Landsifier v1.0: a Python library to estimate likely triggers of mapped landslides

Kamal Rana[1,2,3], Nishant Malik[4], and Ugur Ozturk[1,3]

[1]Helmholtz-Centre Potsdam–GFZ German Research Centre for Geosciences, Potsdam, Germany.
[2]Chester F. Carlson Center for Imaging Science, Rochester Institute of Technology, Rochester, NY, USA.
[3]Institute of Environmental Science and Geography, University of Potsdam, Potsdam, Germany.
[4]School of Mathematical Sciences, Rochester Institute of Technology, Rochester, NY, USA.

Correspondence: Kamal Rana (kr7843@rit.edu)

Abstract.

Landslide hazard models aim at mitigating landslide impact by providing probabilistic forecasting, and the accuracy of these models hinges on landslide databases for model training and testing. Landslide databases at times lack information on the underlying triggering mechanism, making these inventories almost unusable in hazard models. We developed a Python-based unique library, `landsifier`, that contains three different Machine-Learning frameworks for assessing the likely triggering mechanisms of individual landslides or entire inventories based on landslide geometry. Two of these methods only use the 2D landslide planforms, and the third utilizes the 3D shape of landslides relying on an underlying Digital Elevation Model (DEM). The base method extracts geometric properties of landslide polygons as a feature space for the shallow learner—Random Forest (RF). An alternative method relies on landslide-planform images as an input for the deep learning algorithm—Convolutional Neural Network (CNN). The last framework extracts topological properties of 3D landslides through Topological Data Analysis (TDA) and then feeds these properties as a feature space to the Random Forest classifier. We tested all three interchangeable methods on several inventories with known triggers spread over the Japanese archipelago. To demonstrate the effectiveness of developed methods, we used two testing configurations. The first configuration merges all the available data for the k-fold cross-validation, whereas the second configuration excludes one inventory during the training phase to use as the sole testing inventory. Our geometric features-based method performs satisfactorily, with classification accuracies varying between 67% and 92%. We have introduced a more straightforward but data-intensive CNN alternative, as it inputs only landslide images without manual feature selection. CNN eases the scripting process without losing classification accuracy. Using topological features from 3D landslides (extracted through TDA) in the RF classifier improves classification accuracy by 12% on average. TDA also requires less training data. However, the landscape autocorrelation could easily bias TDA-based classification. Finally, we implemented the three methods on an inventory without any triggering information to showcase a real-world application.

1 Introduction

Landslides are gravitational movements of rock and debris that pose a severe threat to the human environment (Depicker et al., 2021). Hazard models are developed to forecast landslides or to aid in understanding landslide processes to mitigate their

undesired consequences (Lombardo et al., 2020). These models commonly rely on mapped landslides to assess the relevant
landslide causes in combination with landslide triggers, i.e., earthquake and rainfall (Lombardo and Tanyas, 2021; Ozturk et al., 2021; Marin et al., 2020). However, many historical landslide inventories lack information about the triggering mechanism decreasing their potential utility in models (Bíl et al., 2021; Martha et al., 2021). More recent semi-automated satellite-based landslide mappers also often disregard the triggering information (Behling et al., 2014, 2016; Ghorbanzadeh et al., 2019), except the event-based inventories—landslide mapping campaigns following a precursory triggering event such as a strong earthquake (Stumpf and Kerle, 2011; Gorum et al., 2014). Using landslide inventories with missing triggers could introduce biases as it is possible to accidentally use an earthquake-triggered inventory to assess rainfall-induced landslide hazards and vice-versa. Hence, classifying the trigger of entire landslide inventories or mapped individual landslides would enhance the usability of newly acquired and historical inventories in landslide models (Guzzetti et al., 2012).

Landslide planforms are used to estimate the mobilized landslide volume, for example, estimating the potential sediment budget of a large landslide triggering events (Malamud et al., 2004; Fan et al., 2012). This type of scaling relationship between the area of landslide planforms to mobilized landslide volume allows comparing the impact of different landslide triggers, such as human versus earthquakes, in terms of the landslides triggered influence on landscape (Tanyaş et al., 2022). However, this area-volume scaling depends on the triggering mechanism of landslides. For example, an earthquake-triggered landslide has a different area-volume relationship than a rainfall-induced landslide. Hence, extracting the landslide triggers information could enhance the estimation capacity of landslide volumes (Moreno et al., 2022) and also help predict the size of co-seismic landslides for a given earthquake (Lombardo et al., 2021). Also, when the exact trigger is known, observed landslides help assess earthquakes' ground motion patterns when no seismic observation is available (Lombardo et al., 2019).

Landslides with the same trigger morphologically cluster, for example, covering narrowly the available statistical variability of hillslope angles in a study region (e.g., Jones et al., 2021) and, thus, could have characteristic shapes reflecting their triggering mechanism, for instance, by having similar area and perimeter ratio, or size (Taylor et al., 2018; Samia et al., 2017). We developed a binary classifier that groups landslides either as earthquake-triggered or rainfall-induced based on this hypothesis (Rana et al., 2021). This initial model demonstrated that the landslides with an identical trigger indeed exhibit similar geometric properties. Thus, finding the trigger of landslides is a classification problem, and one can employ machine learning tools to carry out automated classification of landslide triggers. In each classification problem, the principal idea is to construct a classifier based on training samples and evaluate its performance on testing samples. The classifier predicts the class y corresponding to the input sample x. These input samples x can be one-dimensional vectors or images; for instance, in a soil classification problem (e.g., Bhattacharya and Solomatine, 2006), x is a one-dimensional vector, and in any image classification problem, x is an image (2D or multi-dimensional matrix) (Domingos, 2012).

Our preliminary model (Rana et al., 2021) can classify landslide triggers by only using the geometric properties of landslide polygons. Here, we introduce two additional methods for landslide trigger classification. In one new method, we treated landslide polygons as images, and these images are fed as the sole predictor to a deep learner—Convolutional Neural Networks (CNN). Treating landslide polygons as images eases the workflow as an image already resembles some of the geometric features of the first method. Both these methods rely on two-dimensional (2D) landslide planforms, ignoring the three-dimensional

(3D) shapes of real-world landslides. In another approach, we included the 3D shapes of landslides by incorporating the elevation of landslides via a Digital Elevation Model (DEM). In this approach, we extracted the topological features of these 3D shapes using a recently developed technique known as Topological Data Analysis (TDA). These topology-based features are input to the decision tree-based shallow learner as in the first method. We included the TDA-based model considering its potential to handle other relevant classification problems in future versions of our tool, e.g., classifying landslide types (Cruden and Varnes, 1996; Varnes, 1978). Above listed methods could be used independently following similar script streams.

This study also introduces a new Python library, `landsifier`, that classifies the trigger of landslides, individually or as a whole, in an inventory, where the landslide source mechanism is undocumented. `Landsifier` is the first-ever library built for estimating likely triggers of mapped landslides, the methods used in this library to find landslides' triggers are new. Two of these methods are introduced in this paper for the first time, while the third was published in our preliminary work (Rana et al., 2021). The library consists of three different machine learning-based methods mentioned above; we elaborate on these methods in section 3. Various functionalities of the library are described in Appendix B; where we also list several supporting functions to calculate landslide polygons' geometric properties, convert landslide polygons' shape to a binary-scale image, download a Digital Elevation Model (DEM) corresponding to inventory location, and evaluate the diagnostic performance of the final classification. To demonstrate the efficacy of the developed methods, we apply each to six landslide inventories with known triggers spread over the Japanese archipelago and document our findings in section 4. In section 6, we further highlight the weaknesses of each method to ease choosing the suitable classifier for the various applications.

2 Data

In this work, we used seven landslide inventories spread over the Japanese archipelago (Figure 1). The trigger mechanism of six out of seven landslide inventories are known (Figure 1a–f), whereas the last inventory has no documented triggering information (Figure 1g). We use the last inventory to demonstrate the practical deployment of the final model as this case represents the model's real-world usage. Out of six landslide inventories, three inventories are earthquake-triggered (Figure 1d–f) that are associated with the 2018 $M_W 6.6$ Hokkaido Eastern Iburi (3256 landslides); the 2008 $M_W 6.9$ Iwate–Miyagi Nairiku (4160 landslides), and the 2004 $M_W 6.6$ Niigata (8780 landslides). The remaining three are rainfall-induced (Figure 1a–c), and these are associated with the 2017 Fukuoka-northern Kyushu torrential rainfall disaster (1924 landslides), the 2018 Saka-Japan floods (2817 landslides), and Kumamoto inventory (5564 landslides) that is collected over 1992–2012—not associated with any particular event.

The Geospatial Information Authority of Japan (GSI) is the source of the Hokkaido Eastern Iburi earthquake (September 2018), Fukuoka rainfall (July 2017), and Saka rainfall (July 2018) inventories. The source of the other two coseismic inventories—Iwata and Niigata—is the global repository created by Schmitt et al. (2018). The remaining two inventories from the Kumamoto region are provided by Japan's National Research Institute for Earth Science and Disaster Resilience (NIED). The first inventory from Kumamoto is associated with rainfall (Figure 1b), whereas the second inventory is without any trig-
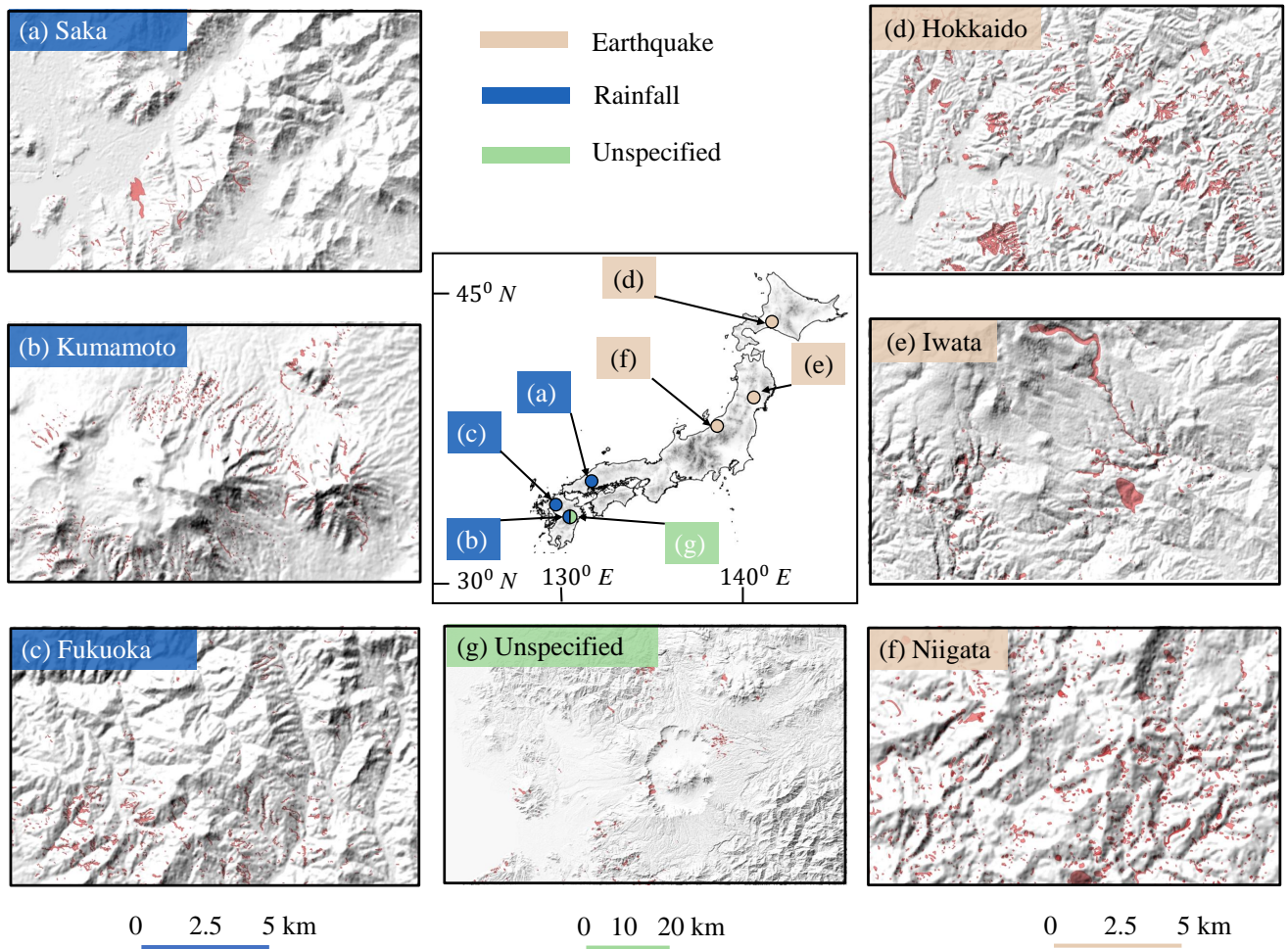
Figure 1. The seven landslide inventories used in this work are spread over Japan, and their geographical locations are shown on the Country's map at the center of the figure. (a)–(g) Shows the subset of landslide polygons highlighted by red color on local hillshades. (a)–(c) Rainfall-induced inventories; (d)–(f) coseismic inventories; (g) undocumented "Kumamoto unspecified" inventory.

gering information (Figure 1g). From hereafter, we refer to this second inventory as "Kumamoto unspecified" (it consists of 612 landslides with unknown triggers).

The TDA-based method uses elevation data to obtain the 3D shapes of landslides from their 2D planforms. We use the *Shuttle Radar Topography Mission* (SRTM) Digital Elevation Model (DEM) data that comes with a spatial resolution of approximately 30 meters. The SRTM data is freely available from `https://www2.jpl.nasa.gov/srtm/` by manually selecting the tiles which correspond to topographic quadrangles. Each tile covers 1 degree of both latitude and longitude region. The `landsifier` library automatically downloads the corresponding tile(s) covering the region of the used landslide inventory (explained further in Appendix B).

3 Methods

In our preliminary study (Rana et al., 2021), we introduced a method that can classify landslide triggers by only using geometric features of landslide planforms. This initial model constitutes the first method in `landsifier` library, and for continuity, we briefly describe it in section 3.1. In this paper, we further diversify our initial model and introduce two new methods, one based on the topological features of 3D shapes of landslides computed using TDA; described in section 3.2. The other new method uses CNN to carry out an image-based classification of landslide triggers; see section 3.3. We anticipate that the variety of methods and corresponding Python library presented here would allow researchers to perform this analysis seamlessly.

3.1 First method: geometric features based classification

In the first method, we used the geometric properties of 2D landslide polygons for the classification. We explored several geometric properties of landslide polygons (e.g., Figure 2). Using a combination of feature selection methods and feature importance analysis, for instance, removing highly correlated features, we choose the seven geometric properties of polygons that lead to optimum results. These geometric features are area A, perimeter P, convex hull based measure $C_h = \dfrac{A}{A_c}$, where, A_c is the area of the convex hull fitted to the polygon (hereafter, we will refer C_h as convex hull measure), the ratio of area to perimeter $\dfrac{A}{P}$, the width of the minimum area bounding box W, minor axis s_m, and eccentricity of the fitted ellipse e having area A and perimeter P. All these seven geometric features are calculated using the Python library, `shapely` (Gillies, 2013). The feature vector ($[A, P, C_h, W, s_m, \dfrac{A}{P}, e]$) is input variable to machine learning algorithm—random forest (described in the Appendix A). Further details of the method can be found in (Rana et al., 2021).

In Rana et al. (2021), we analyzed the distributions of geometric properties of the earthquake and rainfall polygons and found geometric dissimilarities between earthquake and rainfall polygons' shapes. Earthquakes polygons are more likely to have a compact shape (as measured by convex hull-based measure) than rainfall polygons. Moreover, earthquake polygons have more chances to have a larger area (A), perimeter (P), the ratio of the area to the perimeter ($\dfrac{A}{P}$), and minimum width (W) than rainfall polygons. In contrast, rainfall polygons have a larger eccentricity (e) than earthquake polygons of an ellipse fitted to the polygon. Rainfall polygons are more sinuous in shape leading to the smaller minor axis and larger major axis leading to the larger eccentricity of the ellipse fitted to the polygon (Rana et al., 2021).

3.2 Second method: topological features based classification

In the second method, we used the 3D shapes of landslides by incorporating the elevation data of the landslide regions. We
125   extracted geometrical and topological properties of a landslides' 3D shapes using Topological Data Analysis (TDA) and then
used these properties as a feature space for the machine learning algorithm—random forest (described in the Appendix A). The
topological properties of the landslide's 3D shape extracted using DEM provide additional insights into the landslide triggers,
which might further improve the accuracy of the landslide trigger classification. We converted the 2D landslide polygons to
3D landslide polygons using interpolation of 30 meters' elevation data (DEM) around the bounding box of landslides. We
130   took only the elevation data within the landslide polygons to preserve the geometric shape of the landslides (Figure 3). We
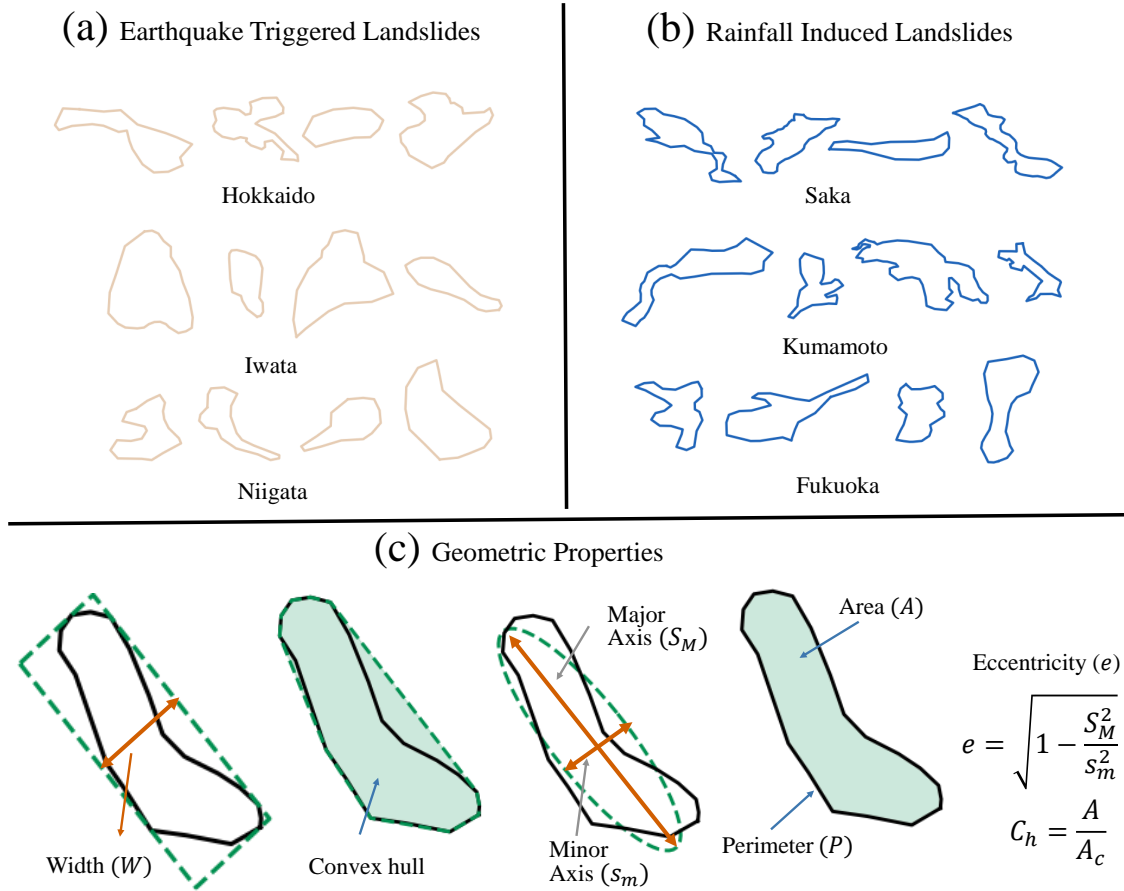
Figure 2. Sample landslide planforms from all six known triggered inventories (a) Earthquake triggered inventories, (b) Rainfall induced inventories. (c) geometric properties of landslide polygon (from left to right): width (W) of the minimum area bounding box fitted to polygon, convex hull based measure (C_h), minor(s_m), and major axis(S_M) of an ellipse fitted to polygon having area A and perimeter P, area (A) and perimeter (P) of the polygon.

explored various TDA features to quantify the 3D shapes of landslides using the Python library, `giotto-tda` (Tauzin et al., 2021). Using random forest feature importance analysis, we selected the top ten most relevant features, as irrelevant features increase the complexity of the model and are ineffective in improving the classification results. These selected relevant features constitute the input variables for the random forest classifier.

135   Topological Data Analysis (TDA) provides a gamut of metrics to quantify the multidimensional shape of data by applying techniques of algebraic topology (Carlsson, 2009). These metrics could also serve as a feature space for machine learning algorithms to solve classification problems, e.g., the classification of manifolds or complex geometric shapes. The central
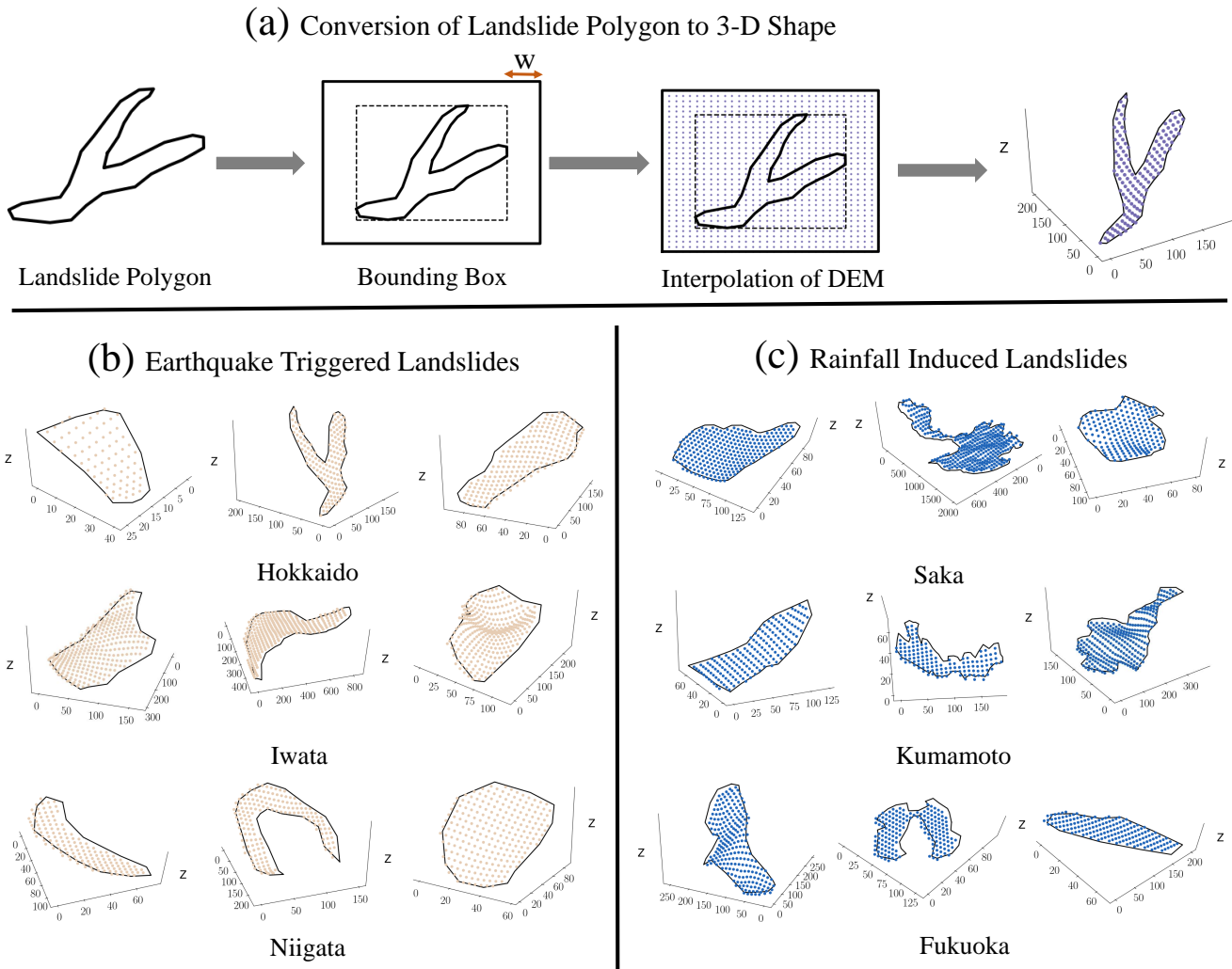
Figure 3. Sample $3D$ landslides from six known triggered inventories, (a) flow chart of conversion of $2D$ landslide planforms to $3D$ landslide shape. (b) Earthquake triggered $3D$ landslide samples, (c) rainfall induced landslide $3D$ samples. The $2D$ landslide planforms converted to $3D$ landslide shapes by using the elevation of landslides through a Digital Elevation Model (DEM).

idea of TDA is persistent homology that identifies persistent geometric features in the data; it uses simplicial complexes to extract topological features from the point cloud data. A simplicial complex is a collection of simplexes and building blocks of higher dimensional counterparts of a graph. For example, a point is a 0-dimensional simplex, an edge which is a connection between two points is 1-dimensional simplex, a filled triangle formed by connecting three non-linear points is a 2-dimensional simplex. In general, an n-dimensional simplex is formed by connecting n+1 affinely independent points (Munch, 2017; Garin and Tauzin, 2019).

Generally, in TDA, one constructs a simplicial complex by the Vietoris-Rips complex method, where one chooses a parameter $\epsilon > 0$ to find the structure present in the data. For each pair of points (x, y) in the point cloud data, add an edge between x and y if euclidean distance (d) between x and y is less than ϵ. For a n-dimensional simplex, distance between each pair of $n + 1$ affinely independent points should be less than ϵ ($d(x, y) < \epsilon$). Each value of ϵ provides a set of simplexes representing a data structure. Different values of ϵ could lead to a different structure in data. To get the complete information about the structures present in the data, all the possible values of ϵ are used, creating a sequence of simplicial complex (this process is called filtration, Figure 4a-g).

Homology measures particular structures present in the data providing valuable information about the geometrical and topological properties of the data. For example 0-dimensional homology captures connected components or clusters, 1-dimensional homology measures loops, 2-dimensional homology measures voids (Munch, 2017; Hensel et al., 2021). Structures like connected components, holes, and voids originate (birth) and disappear (death) with a change in the value of ϵ. A persistence diagram, shown in Figure 4(h); documents the birth and death information of these structures. Using the birth and death information of clusters, holes, and voids present in the persistence diagram, we can calculate several topological features of the data. We used various topological features to quantify the shape of data such as persistence entropy, average lifetime, number of points, betti curve-based measure, persistence landscape curve-based measure, Wasserstein amplitude, Bottleneck amplitude, Heat kernel-based measure, and landscape image-based measure. Each topological metric considers different homology dimensions separately.

The above mentioned topological features can be explained using two objects, one the set of $\{(b_i, d_i)\}_{i=1}^{i=N}$ birth-death pair in the persistence diagram; where i and N are the birth-death pair index and the total number of birth-death pairs respectively, and two the elements of lifetime vector $[l_i]_{i=1}^{i=N}$, calculated as difference between death and life of (b_i, d_i) pair ($l_i = d_i - b_i$). Then the number of points is the length of the lifetime vector, whereas Wasserstein and Bottleneck amplitudes are p-norm and ∞-norm of lifetime vector, respectively. Average lifetime and persistence entropy are average and Shannon-entropy of lifetime vector.

Betti and persistence landscape curves based features are calculated from p-norm of discretized betti and persistence landscape curves. Betti curve is a function $B(\epsilon)$ that maps persistence diagram to an integer-valued curve, $B(\epsilon) : \mathbb{R} \to \mathbb{Z}$, it counts the number of (birth, death) pairs at ϵ that satisfy the condition $b_i < \epsilon < d_i$ (Garin and Tauzin, 2019). Whereas, persistence landscape curve is a function $\lambda(k, \epsilon) : \mathbb{R} \to \mathbb{R}_+$, where $\lambda(k, \epsilon) = k_{max}\{f_{b_i, d_i}(\epsilon)\}_{i=1}^{i=n}$, k_{max} is k-th largest value of set of functions defined by $f_{b_i, d_i}(\epsilon) = max\{0, min(\epsilon - b_i, d_i - \epsilon)\}$ for each (b_i, d_i) pair (Bubenik and Dłotko, 2017).

The heat kernel-based feature is calculated using the p-norm of the 2D function discretization obtained using the heat kernel on the persistence diagram. Heat kernel transforms the persistence diagram to a function on $\mathbb{R}^2$ obtained by placing a Gaussian kernel with standard deviation σ to each (birth, death) pair and negative of Gaussian kernel with same standard deviation in the

175   mirror image of (birth, death) pairs across the diagonal (Reininghaus et al., 2015). Whereas persistence image-based measure is calculated using the p-norm of 2D function discretization obtained using the weighted Gaussian kernel on the birth-persistence diagram. Weighted Gaussian kernel transforms birth-persistence diagram to a function on $\mathbb{R}^2$ obtained by placing a weighted Gaussian kernel with standard deviation σ to each (birth, death - birth) pair in birth-persistence diagram (Adams et al., 2017). In the birth-persistence diagram, the y-axis represents the lifetime (death-birth) information of each (birth, death) pair.
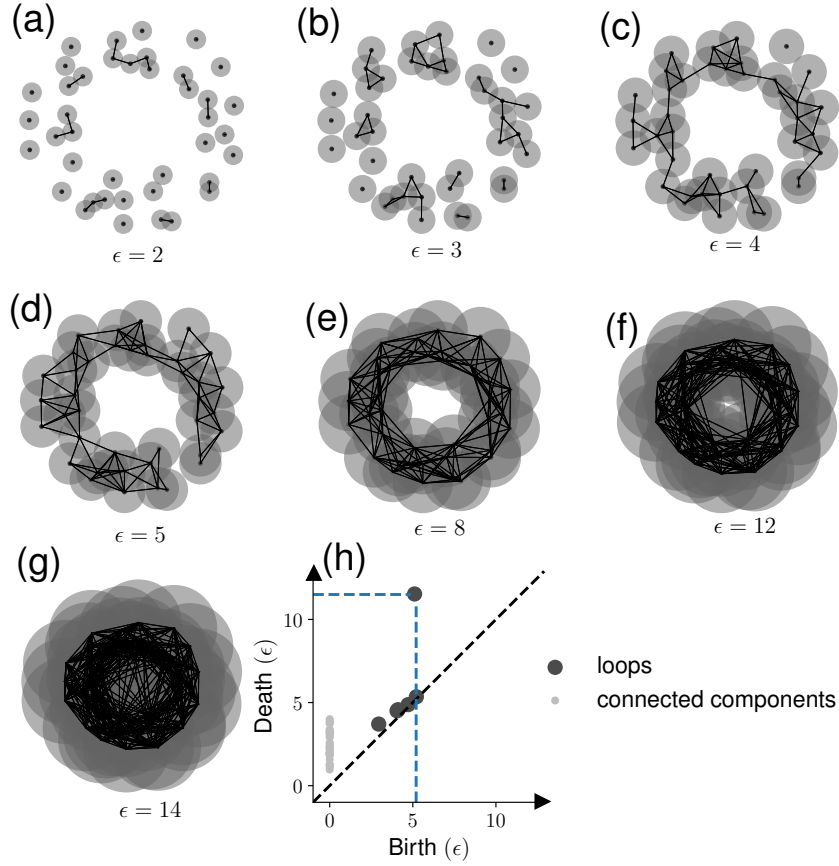
Figure 4. An example of using persistence homology: the data points are sampled from a noisy circle. (a)-(g) As the disk's radius increases ($\frac{\epsilon}{2}$), persistence homology captures various structures in the data. (h) The origin (birth) and disappearance (death) of loops and connected components is shown in the persistence diagram. The biggest loop in the noisy circle data is captured by the data points shown with the blue dotted line in (h).

3.3 Third method: image based classification

In the third method, we used landslide planform images as input to Convolutional Neural Networks (CNN) for the classification. We converted landslide polygons into binary images in a way that preserves the relative shape and structure of the polygons (Figure 5). Then using CNN for landslide triggers classification is straightforward via a simple CNN architecture with 3 convolutional layers and 2 fully connected layers. The input to CNN is a 64×64 binary pixel image, and the output is the probability of the input image belonging to one of the landslide trigger classes.

Convolutional Neural Networks (CNNs) are a class of artificial neural networks that are effective for various applications, such as image classification and object detection (Li et al., 2014; Guo et al., 2017; Albawi et al., 2017). The CNN architecture for classification problems consists of the input, hidden, and output layers (as shown in Figure 6). The input layer consists of the input data to CNN, an image of a landslide polygon in our application. The hidden layer primarily contains convolutional layers, max-pooling, and fully connected layers. Finally, the output layer provides the probability of input data belonging to an output class—rainfall-induced or coseismic.

Convolutional layers are the fundamental component of CNN that uses kernels (matrix of learnable parameters) to perform convolutions operations on the input. The resulting output of the convolution operation is called a feature map that learns the feature representation of the input data (Yamashita et al., 2018). Each neuron in a feature map captures the antecedent layer's local characteristics by convolution of kernels with the previous layer's feature maps (Guo et al., 2017). However, increasing convolutional layers could lead to over-parametrization and increase model complexity and, thus, over-fitting. One of the ways to avoid the issue is to use pooling layers that reduce feature maps dimension and the number of neurons in the output layer of CNN's (Yamashita et al., 2018; Guo et al., 2017). We used max-pooling layers of $n \times n$ ($n = 2$) size that takes a patch of size $n \times n$ from a feature map and produces one-value corresponding to that patch, and the pooling layer itself is free from parameters (Li et al., 2014).

Activation functions in CNN's capture the non-linear relationship between the input data and its output class. We used ReLu for the hidden layer neurons activation functions as past studies have proved that ReLu improves classification results and learning speed (Li et al., 2014; Krizhevsky et al., 2012). The output of ReLu activation function is $f(x) = max(0, x)$, here x means the output of a neuron (Li et al., 2014). For the output layer, we used the softmax activation function. The softmax activation function calculates the output probabilities of the input sample belonging to each class in the last layer of CNN. The class probabilities are calculated as

$$P_i = \frac{\exp z_i}{\sum_{j=1}^{j=m} \exp z_j}, \tag{1}$$

where z_i is the output from last layer of CNN corresponding to i class and m is the number of classes (in our case, $m = 2$).

Fully connected layers (FCC) work as a classification layer for CNNs and comes after the convolutional layers. All layers in FCC are fully connected which means each neuron in a layer is connected to every neuron in the next layer of FCC (Albawi et al., 2017; Guo et al., 2017). In classification problems, the last layer of the FCC layer gives the probabilities of the input image to belong to one of the output classes with the help of the softmax activation function (Eq. 1). The output predicted

probabilities of the input sample are used in a loss function that evaluates how well the model works for classifying the class of the input image dataset. We used the cross-entropy loss function that measures the difference between actual and predicted probability distribution. The cross-entropy loss function for a sample is defined as: $-\sum_{i=1}^{i=m} y_i log(\hat{y}_i)$, where m is the total number of classes, y_i ($\hat{y}_i$) is actual (predicted) probability corresponding to class i. If i is actual class of the input sample then $y_i = 1$, otherwise $y_i = 0$. In the case of binary classification $m = 2$. The sample's output probabilities are a function of parameters used in convolution kernels and FCC layers to connect neurons in one layer to the next layer. These parameters are altered iteratively using the back-propagation algorithm and stochastic gradient method to increase the probability of samples belonging to the actual class and thus, minimize the loss (Aurisano et al., 2016).

4 Landsifier model evaluation
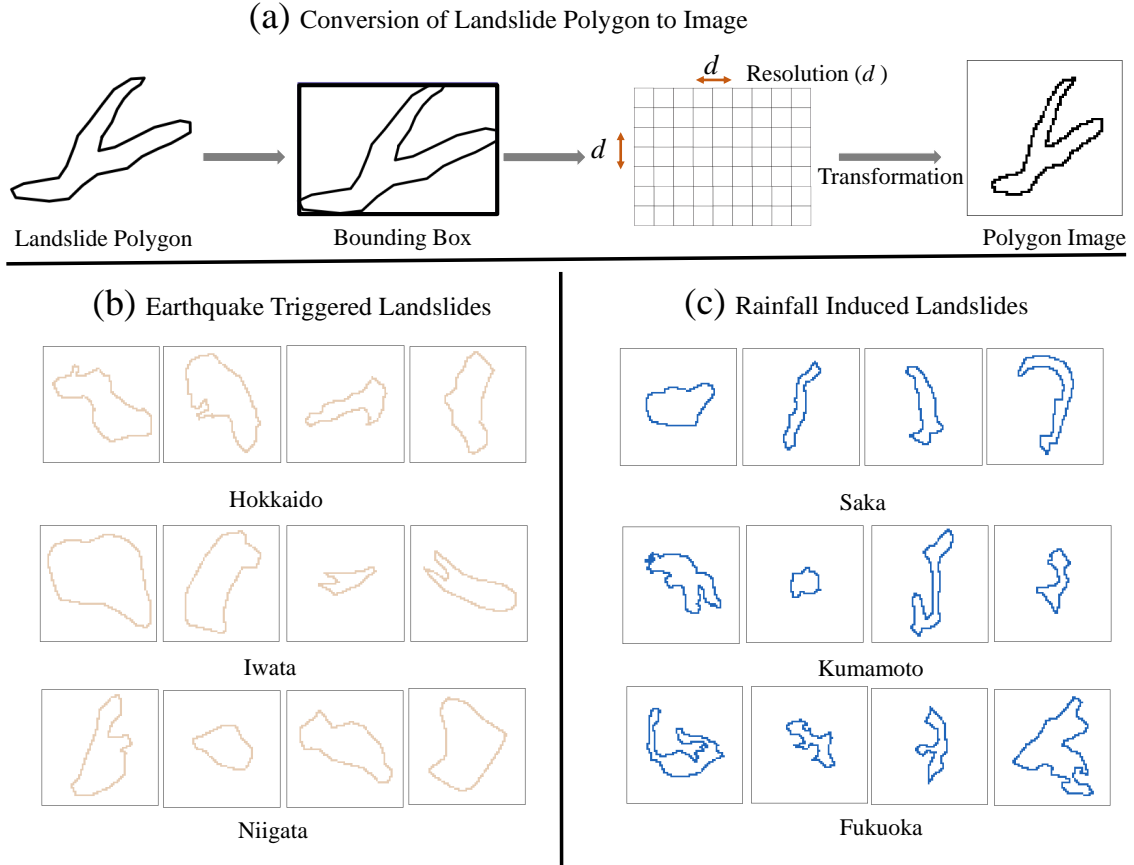
Figure 5. Sample input images for the image-based classification. (a) Flow chart of converting landslide planforms to a landslide polygon image. (b) Earthquake-triggered landslide image samples. (c) Rainfall-induced landslide image samples.

We used two different testing configurations to evaluate the efficacy of our methods. Finding the triggers of individual landslides irrespective of their inventories is the first testing configuration. Here, we combined all the known trigger landslides from all six known triggered inventories and then split the combined landslides data into various training and testing sets following the k-fold cross-validation framework. In this testing configuration, landslides in each training and testing set are from all six landslide inventories. The second testing configuration finds the trigger of landslide inventories itself. We used all the possible combinations to train the algorithm on five known trigger inventories and test it on the sixth inventory. In this second testing configuration, landslides in the testing set are from a single inventory. Note that there are seven inventories in the analyzed data set, and six have known triggers. The analysis of this seventh inventory (Kumamoto unspecified) with unknown triggers is presented in the section 6.

4.1 Evaluation of the first method (geometric features based classification)

Combining all the landslide inventories with known triggers lead to $26,501$ samples (n_{total}), out of which $16,196$ are earthquake-triggered landslides ($n_{earthquake}$) and $10,305$ are rainfall-induced landslides ($n_{rainfall}$). As the number of earthquake-triggered landslides is much larger than the number of rainfall-induced landslides, we use equal numbers of each trigger class to avoid any class-imbalance problems. To avoid selection bias and overfitting, we apply 10-fold cross-validation. k-fold cross-validation splits the combined classes dataset into k random subsets where each iteration of cross-validation, $k-1$ folds are used for train-
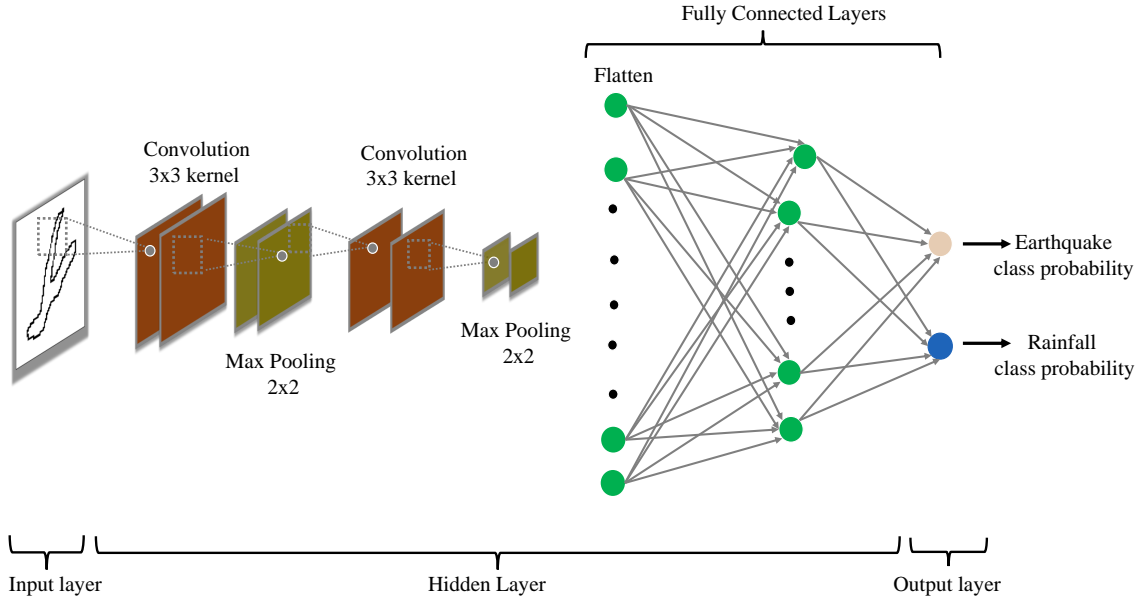
Figure 6. The figure shows the Convolutions Neural Network (CNN) architecture used in the image-based method. The input of CNN is a binary scale landslide image, and the output of CNN is the probability of a landslide image belonging to an earthquake or rainfall-induced class.

ing and the remaining fold for testing. We use $20,610$ samples ($n_{rainfall} = n_{earthquake} = 10,305$) for cross-validation and to get generalizable results we employ 1000 runs of cross-validation. In each run of cross-validation we randomly select $10,305$ earthquake samples from $16,196$ earthquake landslides. We achieved $86.15 \pm 0.22\%$ classification accuracy for earthquake, $85.29 \pm 0.19\%$ for rainfall, and $85.73 \pm 0.16\%$ as the mean classification accuracy.

For the second split configuration, we trained the random forest classifier on five inventories and tested it on the sixth inventory. For earthquake triggered inventories the method achieved classification accuracy of $66.62 \pm 0.65\%$, $75.59 \pm 0.34\%$ and $85.22 \pm 0.20\%$ for the Hokkaido ($n_{train} = 20,610, n_{test} = 3,256$), Iwata ($n_{train} = 20,610, n_{test} = 4,160$) and Niigata ($n_{train} = 14,832, n_{test} = 8,780$) inventories (for geographical locations of these inventories see Figure 1). For rainfall induced inventories, we achieved classification accuracy of $83.63 \pm 0.41\%$, $69.40 \pm 0.61\%$ and $92.12 \pm 0.25\%$ for Kumamoto ($n_{train} = 9,482, n_{test} = 5,564$), Fukuoka($n_{train} = 16,762, n_{test} = 1,924$) and Saka ($n_{train} = 14,946, n_{test} = 2,817$) region. In each one of the the case we took equal number of earthquake and rainfall triggered landslide samples to avoid any class imbalance issues ($n_{earthquake} = n_{rainfall}$). The low standard deviation in classification accuracy shows that results are stable with change in training samples.

4.2 Evaluation of the second method (topological features based classification)

In the first test and training set split configuration, as in Section 4.1, we used $n_{total}= 20,610$ (total number of samples), $n_{earthquake}=10,305$ (number of earthquake-triggered samples) and $n_{rainfall} =10,305$ (number of rainfall-induced samples), keeping numbers of each trigger class equal to avoid class imbalance. We first identify the top ten relevant topological features out of thirty features, employing 1000 runs of 10-fold cross-validation of random forest. Using these top ten relevant topological features as the feature space for the random forest classifier, we carry out 1000 runs of 10-fold cross-validation to get generalized classification accuracy. The method achieved above 94% classification accuracy for earthquake, rainfall, and mean class classification.

In the second split configuration, this method achieves above 90% accuracy for the Iwata, Niigata, Kumamoto, and Saka inventories. For the Hokkaido and Fukuoka region, the method achieves above 80% classification accuracy (see Figure 7). The number of training and testing samples for each case is the same as in Section 4.1.

4.3 Evaluation of the third method (image based classification)

As explained above in section 3.3 we removed large landslides from the analysis leading to $n_{total} = 24,311$, $n_{earthquake} = 14,892$, and $n_{rainfall} = 9,419$. We used an equal number of training samples of the coseismic and rainfall-induced landslides to avoid any class imbalance issues. We used 100 runs of different test and training sets instead of different runs of 10 fold cross-validation as convolutional neural networks are computationally expensive. The method achieved above 85% classification accuracy for earthquake, rainfall, and mean class classification.

For the second split configuration, the method achieved above 80% accuracy for the Saka region ($n_{train} = 13,738, n_{test} = 2,550$). For the Niigata ($n_{train} = 12,780, n_{test} = 8,502$), Kumamoto ($n_{train} = 8,276, n_{test} = 5,281$) and Fukuoka ($n_{train} = 15,662, n_{test} = 1,588$) region the method achieves accuracy of above 70%. The Method achieves 67% accuracy for the

270   Hokkaido inventory ($n_{train} = 18,838, n_{test} = 2,431$). In each one of the the cases, we took equal number of earthquake and rainfall induced landslide samples to avoid class imbalance issues.

5   `Landsifier` library

One of the main aims of this paper is to introduce `Landsifier`, a Python library we built to provide the landslide re-
275   search community with a user-friendly computational package to implement the methods described above. At the moment, we have made the code available on the corresponding author's GitHub: `https://github.com/kamalrana7843/ Landsifier.git`. Furthermore, we published the `Landsifier` library under an open-source license in a way that it is accessible via the python terminal using the import command. In Appendix B we provide details of the library and brief descriptions of the available functionalities. Apart from three different methods for landslide trigger classification, the library
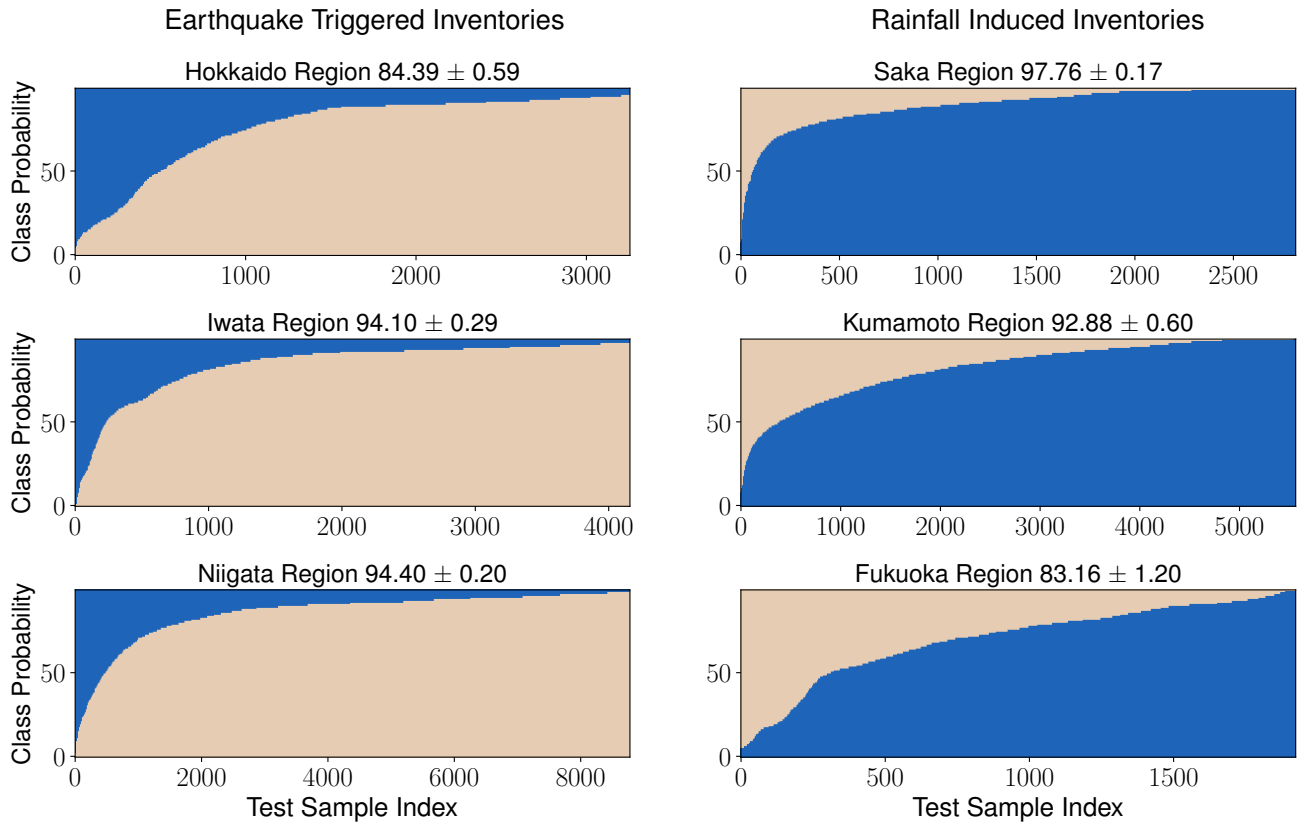
Figure 7. The topological features-based method (second method) accuracies for all the six known triggered inventories. The model is trained on five inventories in each case and tested on the sixth inventory. The y-axis in the plot shows the probability of landslides belonging to the earthquake and rainfall class, and the x-axis shows the sample index of landslides.

also contains other useful functions like calculating geometric properties of landslide polygons, converting polygons to binary scale images, downloading DEM corresponding to an inventory region, and converting 2D landslide polygon to 3D landslide shape (see Figure 3a and 5a). Please refer to Appendix B for further details about the library functions (Figure B1). Each of the three methods used in the library is simple to use and only requires polygon shapefiles as input. Also, the computation process is relatively fast; for example, the geometric, image, and topological features-based method takes less than 5, 15, and 45 minutes for training on 20,000 landslides (equal earthquake and rainfall samples) on a windows machine with 16 GB of RAM (Random-Access Memory) using only landslide shapefiles as input. Moreover, none of the methods requires a GPU (Graphics Processing Unit).

6 Discussion

The geometric properties of landslides can provide information about their trigger (Taylor et al., 2018). Our preliminary work on landslide trigger classification demonstrated that landslides with identical triggers share similar geometric properties, which could be exploited to classify landslide triggers—see the publication Rana et al. (2021) and briefly reproduced results here in sections 3.1, 4.1 and Appendix B1. In this work, we further expanded our initial approach by adding two additional methods for landslide triggers classification and a Python library `Landsifier` to implement them. One of these two new methods uses 3D shapes of landslides for their trigger classification by incorporating the elevation information. We compute topological features of these 3D shapes using Topological Data Analysis (TDA) and use the features as an input to a machine learning-based algorithm—random forest. The other method uses binary scale landslide polygon images as an input to Convolutional Neural Networks (CNN) for the classification. Using six landslide inventories spread over the Japanese archipelago, we showed that each method exhibit strong performance to classify landslide triggers. However, each method has its strengths and limitations that primarily depend on training and testing landslide data quality, quantity, and location. We explained each methods' strengths and limitations in different conditions in this section. Before providing some hints about potential future work and opportunities that could arise from using `Landsifier` library, here, we also present and discuss the results of each three method on the seventh Kumamoto unspecified inventory.

The landslide data quality depends on the data acquiring technique; e.g., landslide data obtained using aerial or satellite images are much higher quality than the data acquired via field campaigns. Geologists collect landslide data acquired via field campaigns, and by nature, such inventories tend to fail to represent the smaller landslides and cover the larger landslides (Ozturk et al., 2020). Whereas landslides inventories acquired via aerial or satellite images cover both small and larger landslides and are called complete inventories as they adequately capture landslides of various sizes in their respective study area, e.g., see (Schmitt et al., 2018). The performance of developed methods depends on landslides data quality and without similar data quality in training and testing set the accuracy of classification techniques could be insufficient to conclude the trigger of landslide inventory and also might lead to biases. Training the geometric feature-based and image-based methods on landslide planforms with landslide data acquired via satellite or aerial images and testing on data acquired via field campaign or vice-versa could lead to biases in landslide classification results. . The methods based on landslide planforms shape consider the

Inventory Region	Geometric features based method (%)	Topological features based method (%)	Image based method (%)
Hokkaido	67	**84**	68
Iwata	76	**94**	67
Niigata	85	**94**	77
Saka	92	**98**	88
Kumamoto	84	**92**	78
Fukuoka	69	**83**	70

Table 1. The table shows landslide classification results using the three methods. The model is evaluated on all possible training set combinations of the five inventories and tested on the sixth inventory.

area and perimeter as the most important features and rely on the information that coseismic landslides are generally larger than rainfall-induced landslides (Rana et al., 2021) (e.g., Taylor et al., 2018; Tanyaş et al., 2021). So, a testing inventory triggered by rainfall but lacks smaller landslides due to field campaign acquisition technique could be classified as earthquake-triggered—given that training inventories are satellite or aerial image-based. We recommend using similar field campaign acquired inventories with known triggers to train the models for more accurate classification in such a scenario. Another option is to sample landslides from the satellite or aerial image-based inventories that resemble the size distribution of the testing data acquired via field campaign. This shortcoming motivated us to offer another alternative solution relying on topological analysis of 3D shapes of landslides.

Landslides are 3D shapes; thus, using 3D shapes of landslides instead of 2D could provide additional information related to the landslide morphology. Consequently, a 3D landslide shape-based method might elevate classification accuracy, especially in regions without proper training and testing data of similar quality. We use TDA, a method rooted in algebraic topology, to compute topological features of a landslides' 3D shapes to classify landslide triggers. TDA based method extracts topological information along with geometric information of landslide shape. Whereas, geometric features based method and likely Image based method use only geometric information of the landslide shape for landslide classification. We expect TDA based method will provide best landslide trigger classification results. In Table 1 one can observe that the TDA-based method indeed performs better than the other two methods. However, TDA-based measures encode landslide morphology; hence, if testing and training inventories share similarities in the geomorphology of the studied regions (spatial autocorrelation) (Oksanen and Sarjakoski, 2005), then the trigger prediction is highly influenced by training inventory. Geometric features and image-based methods are less sensitive to the geomorphological similarities between the training and testing landslide inventories, as these only use the 2D landslide planforms. Although the image-based performs satisfactorily only when adequate large training data is available. Hence, we recommend using geometric or topological features-based methods in inventories with limited landslide counts.

We applied each method to classify landslides triggers in the Kumamoto unspecified inventory having an undocumented trigger to demonstrate the real-world application of the `Landsifier` library. Out of 612 landslides in the inventory, the ge-

ometric feature-based and topological feature-based classified 604 and 612 landslides as earthquake-triggered. In comparison, the image-based method uses 164 landslides after removing landslides having width and length greater than 180 meters (see section 3.3 for more details) and classified all of the landslides as seismically triggered. As each method classifies the majority of the landslides as earthquake-triggered, we are confident that earthquake is the most likely trigger for most of the landslides in this inventory. Moreover, the Kumamoto unspecified inventory documents landslides along the rims of the Aso Caldera, the active volcano Mount Aso shakes the surrounding area frequently triggering landslides within its vicinity (Saito et al., 2018). Hence, it is very likely that this inventory is consistent of landslides of cosesimic origin.

Considering the above discussions, in future work, we plan to explore further the sensitivity of our trigger classification methods to spatial autocorrelations. We will also examine the influences of landslide size distributions on each method. Specifically, we plan to classify the trigger of large landslides (Area $> 90,000$ square-meters) as they are the most dangerous landslides and effect huge area by training each method on large landslides training dataset. Moreover, we will consider model transferability to different regions by extensively testing these methods on national landslide inventories, e.g., India, Nepal, Taiwan, and the USA. Our methods could also provide other opportunities. For example, assessing landslide-prone regions as an alternative to landslide susceptibility measure using TDA. Also, TDA could be used to classify landslide types, according to the types described in Cruden and Varnes (1996) and Varnes (1978). Landslide type information plays a crucial role in landslide risk assessment which is usually missed in landslide databases (Loche et al., 2022). We plan to further develop the current version of the `Landsifier` by incorporating a landslide type classifier in the next version (e.g., Amato et al., 2021). This method will be able to find the analogy between an observed landslide and a generic landslide types by Cruden and Varnes (1996).

7 Conclusions

The landslide triggering mechanism is crucial information to develop landslide hazard models, e.g., a landslide hazard model for extreme rainfall incidents requires landslide inventories related to rainfall events only. However, modern automated landslide mappers for continuous monitoring and historical landslide inventories rarely report the landslide triggering mechanism. Missing triggers in the landslide inventories decrease their efficacy for landslide hazard models. In this work, we developed a Python library, `Landsifier`, containing three methods for landslide trigger classification by exploiting landslide planforms and 3D shapes. To develop the first two of these methods, we combined geometric and topological features with machine learning, and in the third method, we used deep learning. The latter two methods are new, i.e., we are reporting them here for the first time.

We use seven landslide inventories spread over the Japanese archipelago. Six among these seven inventories have known triggers, while the seventh inventory has a missing trigger. We applied each method to all possible sets of five training inventories and one testing inventory using six known triggered inventories. Moreover, we took different training and testing sets of landslides by mixing all known triggered landslides inventories following the k-fold cross-validation. The achieved results demonstrate that the methods are robust and capable of classifying triggers of landslide inventories with high accuracy (70%–

95%). To demonstrate the real-world application of our toolbox, we also applied the three methods to the seventh inventory without any triggering information and classified it as an earthquake-triggered inventory.

370   Python-based `Landsifier` library provides a user-friendly computational package to implement the methods described above to the landslide research community. Two of the three methods included in the library are new and introduced here for the first time, while the third method is published in our previous work. To our best of knowledge `Landsifier` is the first python tool developed for landslide triggers classification, and also such a tool does not exist in other programming languages. We anticipate that the landslide research community will find the `Landsifier` library helpful in finding the trigger mechanism

375 of inventories or individual landslides. The presented methods and the library could be deployed in any region of the world with adequate training data from areas with similar climatic and tectonic features. `Landsifier` library also contains useful functions like finding geometric properties of landslides polygon, downloading DEM corresponding to an inventory region, and converting landslide polygon to landslide 3D shape, these elements could be useful for the landslide research community."

Furthermore, methods in `Landsifier` library are easy to use as they require only shapefiles of landslide polygons as input.

380 `Landsifier` is a modular software, we hope the landslide community will further improve the offered tool and expand the available functions for new applications such as classifying landslide types, assessing landslide-prone regions, and other possible usage are listed in the discussion section. At the moment, we have made the code available on the corresponding author's GitHub: `https://github.com/kamalrana7843/Landsifier.git`. Moreover, we published the `Landsifier` library under an open-source license in a way that it is accessible via the python terminal using the import command. In

385 Appendix B, we provide details of the library and brief descriptions of the available functionalities.

Appendix A: Random forest

Random forest (RF) is a decision-tree based ensemble-learning method, a proven and powerful technique for classification and regression (Barnett et al., 2019; Biau, 2012; Biau and Scornet, 2016; Breiman, 2001; Kursa, 2014; Chaudhary et al., 2016; Rodriguez-Galiano et al., 2012). The random forest classifier consists of multiple classifiers, where each classifier bootstraps

390 the training data samples (Breiman, 2001; Liaw et al., 2002). Bootstrapping in each random forest classifier is done by selecting N samples randomly from training samples of size N with replacements. For N training samples bootstrapping N times leads to the approximate selection of $2/3$ of training samples (Azar et al., 2014; Belgiu and Drăguţ, 2016). Hence, each tree in a random forest classifier is trained independently using around $2/3$ of the training samples selected using bootstrapping.

In a binary classifier, as in our case, each parent node q splits into two daughter nodes: right r and left l. Instead of selecting

395 all the p features for node split, a subset of features m ($m=\sqrt{p}$) is selected randomly for each node split (Azar et al., 2014; Okun and Priisalu, 2007). Among m features, one of the features selected for the node split is based on optimizing a criterion. The criterion is called the 'Gini Index,' which measures the features' impurity to the classes. The Gini index of right r and left
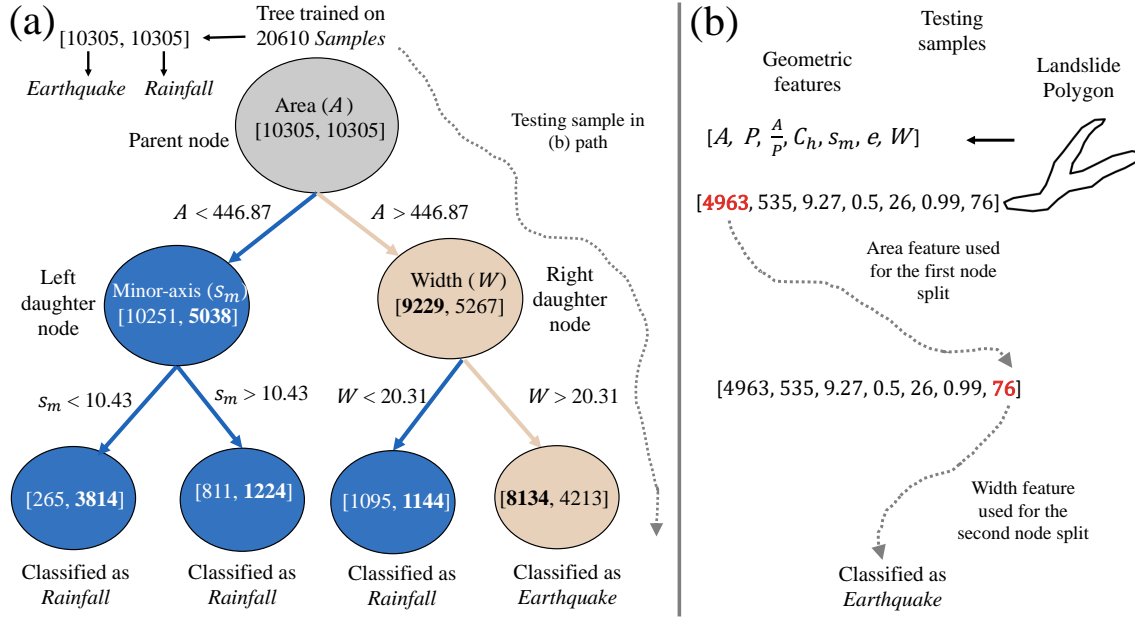
Figure A1. (a) The sample architecture of one of the trees of random forest classifier. The tree is trained on 20610 landslides samples with 10305 each earthquake and rainfall trigger class. Feature vector ($[A, P, C_h, W, s_m, \frac{A}{P}, e]$) represents landslide geometric property corresponding to each landslide sample. For illustration purposes, the tree is grown to only depth three. (b) testing sample of landslide tested on the tree shown in (a). The sample landslide polygon is classified as an earthquake.

l daughter nodes are calculated as:

$$G_r = 1 - P_{r1}^2 - P_{r2}^2 \tag{A1}$$

400
$$G_l = 1 - P_{l1}^2 - P_{l2}^2, \tag{A2}$$

where P_{rj} (P_{lj}) is the probability of samples in the right (left) daughter nodes having class j. The Gini index is calculated for each predictor in the subset of predictors m, and the features that maximize the change in Gini index is chosen for node split. Change in Gini-index is calculated as:

$$\Delta\theta(s_q) = G_q - \rho_{rq}G_r - \rho_{lq}G_l, \tag{A3}$$

405 where ρ_{rq} (ρ_{lq}) are the ratio of the number of data points in daughter nodes r (l) to the total number of points in the parent node q (Kuhn et al., 2013; Zhang and Ma, 2012). The process of splitting nodes continues until a stopping criterion is met, e.g., when no further samples are remaining, or the Gini-index of parent nodes is lower than the daughter nodes.

The steps for constructing trees in the random forest are as follows:

(i) Select bootstrap samples of size N from training samples of size N.

(ii) Randomly select m variables among p variables for the node split.

(iii) Choose one variable among m variables that best split the node according to the Gini-index criterion.

(iv) Continue repeating steps (i) to (iii) until the stopping criterion is met.

For testing, each tree classifier predicts the class of testing sample independently, and the class with majority votes is the final outcome of random forest (Kuhn and Johnson, 2013; Pal, 2005; Arabameri et al., 2021; Belgiu and Drăguţ, 2016). In random forest, bootstrapping training samples selection and random selection of features for a node split reduces the correlation between trees. This technique has proven to improve the predictive power of ensemble learning (Azar et al., 2014). In addition, random forest assigns each feature a score that provides its relative importance (Qi, 2012; Friedman et al., 2001). Features with low relative scores should be discarded as they are neutral to the model accuracy and increase the model complexity.

Appendix B: Details of `Landsifier` library

`Landsifier` is a Python library we built with version 3.6 of Python and the code is available on GitHub: `https://github.com/kamalrana7843/Landsifier.git` (we will publish this library under an open-source license after this manuscript is accepted for publication). On this link, prospective users can also find the list of Python packages used in the library. `Landsifier` contains three methods for landslide trigger classification, and these methods only use shapefiles from landslide inventories (two of these methods use 2D polygon shapes of landslides, while the third method uses the 3D shapes of landslides). This section describes various functions provided in the `Landsifier` library to implement the above methods and Figure B1 summarizes these functions in form of a flowchart. Also, Figure B2 shows a sample output of the `Landsifier`.

B1   Functions for geometric features based classification

Below we list functions to implement the geometric features-based classification, details of the method can be found above in section 3.1 and in our publication (Rana et al., 2021). Note below we have described functions in a form that this method can be used for inventories with unspecified triggers, i.e., unknown ground truth.

`latlon_to_eastnorth(latlon_polydata)`: This function takes polygons data in $(longitude, latitude)$ coordinates as an input and provides polygon data in $(easting, northing)$ coordinates as output to the function. This function is used to get landslide polygons in $(easting, northing)$ coordinates when polygon data in shapely files are in $(longitude, latitude)$ coordinates.

`Calculate_geometric_properties (polygon_shapefile)`: As the name suggests, this function calculates the geometric properties of each of the landslide polygons present in shapefile. This function takes polygons shapefiles (`polygon_shapefile`) as input, converts polygon data into $(easting, northing)$ coordinates if required using the
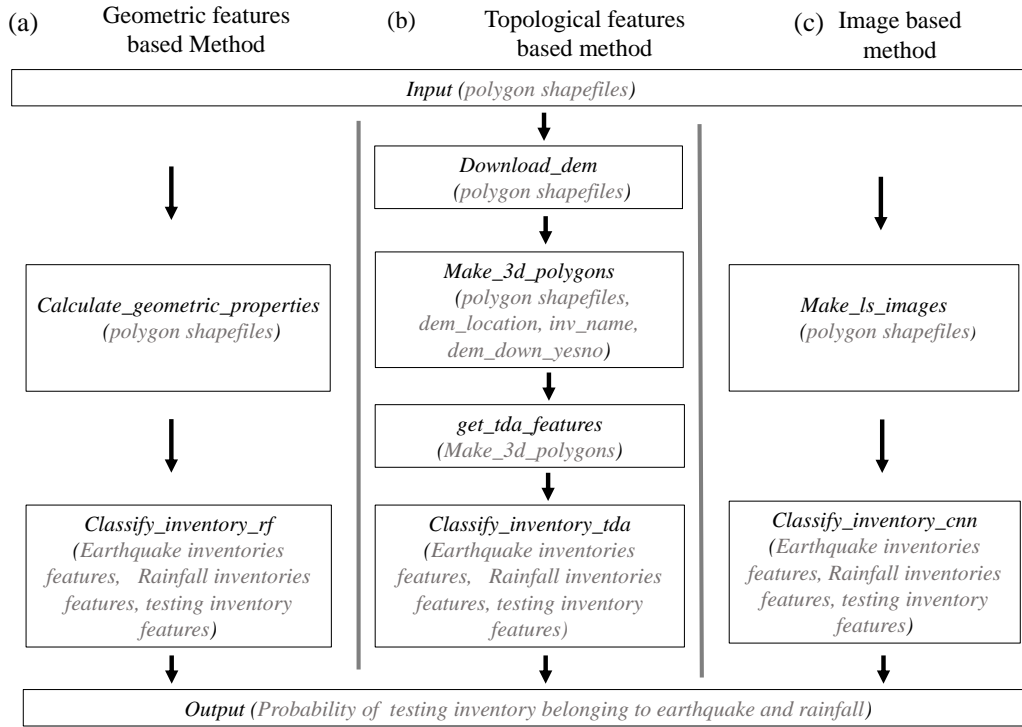
Figure B1. The figure shows the flowchart of implementations of all the three methods using functions and their variables used in the `landsifier` library. All three models use polygon shapefiles as an input to the model and provide the probability of landslide belonging to earthquake and rainfall as an output (a) geometric features based method (b) topological features based method (c) image-based method.

`latlon_to_eastnorth` function, and then provides the geometric properties of polygons as output to the function. For each landslide polygon it calculates a vector($[A, P, C_h, W, s_m, \frac{A}{P}, e]$) containing polygon geometric properties as output to the function. All the geometric properties of the landslide polygon are calculated using the $shapely$ package in Python.

445   `classify_inventory_rf (earthquake_inventory_features, rainfall_inventory_features, test_inventory_features)`: This function takes the earthquake-triggered inventories ($earthquake\_inventory\_features$), rainfall-induced inventories ($rainfall\_inventory\_features$) and testing inventories ($test\_inventory\_features$) geometric features as the input. Within the function, it trains the random forest algorithm on training data containing equal samples of the earthquake and rainfall-induced class. The output of the function is the probability of testing landslides belonging to each

450   trigger class.
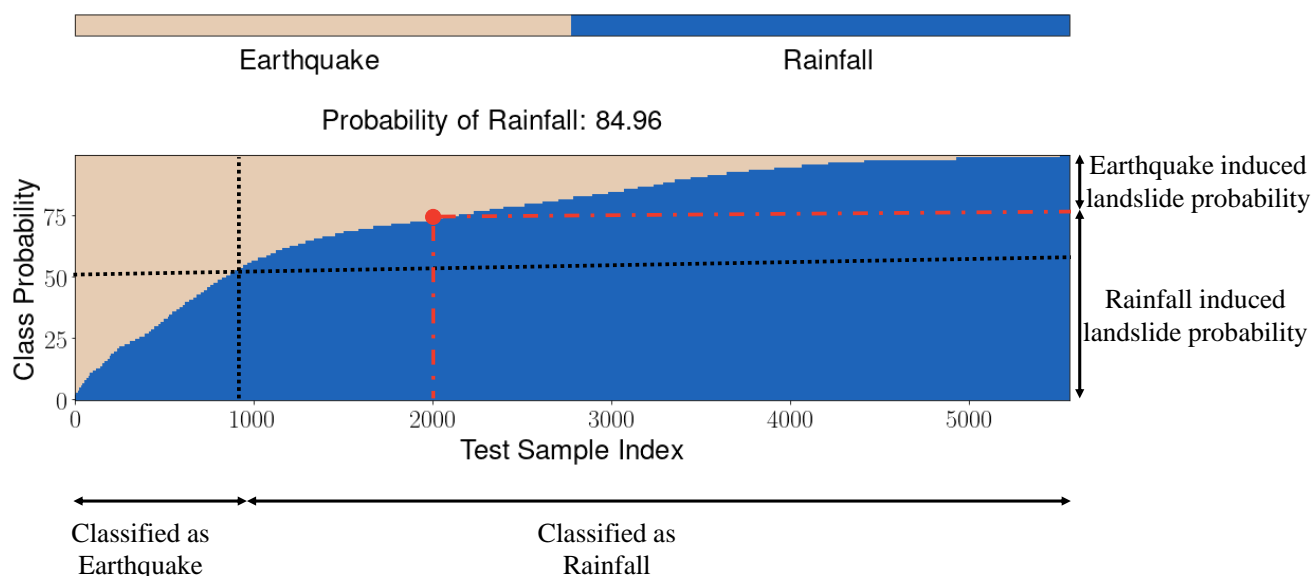
Figure B2. The output of the geometric feature-based method when Kumamoto inventory is testing inventory and the rest five inventories are used as training inventories. Each method in the `landsifier` will produce similar outputs. The y-axis in the plot shows the probability of landslides belonging to the earthquake and rainfall class, and the x-axis shows the sample index of landslides. For each landslide in the testing inventory, all the models give a probability of landslides belonging to earthquake and rainfall-induced classes. The predicted trigger of most of the testing landslides is the probable trigger of the testing inventory.

B2 Functions for topological features based classification

Below we list functions to implement the topological features-based classification, details of the method can be found above in section 3.2. Note below we have described functions in a form that this method can be used for inventories with unspecified triggers, i.e., unknown ground truth.

455

`download_dem (polygon_shapefile)`: This function takes shapefile of landslide polygons as an input and downloads the *Shuttle Radar Topography Mission* digital elevation model (DEM) of resolution 30 meters corresponding to inventory region (Farr et al., 2007). It takes the bounding box over the entire inventory location and calculates the (`minimum latitude, minimum longitude`) and (`maximum latitude, maximum longitude`). Using the elevation

460 package in Python, it downloads the DEM data of a region bounded by `minimum latitude, minimum longitude, maximum latitude, and maximum longitude` coordinates. This function downloads all the tiles (one tile consti-

tutes $1° \times 1°$ region in both latitude and longitude) of the inventory region and combines all the tiles into one file corresponding to one inventory.

465 `make_3d_polygons (polygon_shapefile, dem_location, inv_name, dem_down_yesno )`: This function takes landslide polygon shapefiles (`polygon_shapefile`), DEM path location (`dem_location`), inventory name (`inv_name`) and Boolean parameter (`dem_down_yesno`) as input and provides 3D point cloud data of landslides as output. This function carries out several tasks. First, it downloads the DEM data corresponding to the whole inventory region in path location (`dem_location`) with inventory name (`inv_name`) using `download_dem` function if `dem_down_yesno` is True.

470 If users already have DEM corresponding to inventory in path location (`dem_location`) with inventory name (`inv_name`) then (`dem_down_yesno`) is False. Then corresponding to each landslide polygon it interpolates the DEM data around the bounding box of the polygon. Using the shapely package, the function removes all the interpolated data outside the outline of the landslide polygon and takes elevation data only within the landslide.

475 `get_tda_features (three_d_data)`: This function takes the 3D shape of landslides point cloud data (`three_d_data`) as an input and provides machine learning features corresponding to each 3D landslides as an output to function. This function calculates the persistence diagram using Vietoris Rips persistence for each 3D landslide, and then using the persistence diagram, it calculates the following TDA metrics: persistence entropy, average lifetime, number of points, betti curve based measure, persistence landscape curve based measure, Wasserstein amplitude, Bottleneck amplitude, Heat kernel-based mea-

480 sure, and landscape image-based measure corresponding to each homology dimension–0, 1, and 2. These TDA metrics are used as a feature space for the machine learning algorithms.

`classify_inventory_tda (earthquake_inventory_features, rainfall_inventory_features, test_inventory_features)`: This function takes training earthquake inventory's (`earthquake_inventory_features`),

485 training rainfall inventory's (`rainfall_topological_features`) and testing inventory's (`test_inventory_features`) TDA based features as input to function. Inside the function, it first selects the top 10 features with the highest feature importance using training data. It then combines an equal number of training earthquake and rainfall samples to avoid any class imbalance problem. It trains the random forest algorithm on training data and predicts the probability of testing landslides belonging to each trigger class.

490

B3   Functions for image based classification

Below we list functions to implement the image-based classification, details of the method can be found above in section 3.3. Note below we have described functions in a form that this method can be used for inventories with unspecified triggers, i.e., unknown ground truth.

495

`increase_resolution_polygon (poly_data)`: This function takes a single polygon coordinates data (`poly_data`) in $(easting, northing)$ coordinates as input and increases the number of points between any two adjacent vertexes of the polygon within the function. This function is useful in creating smooth binary scale landslide polygon images. The output of the function is landslide polygons coordinates data having multiple points between the adjacent vertex of polygons.

500

`make_ls_images (polygon_shapefile)`: This function takes polygon shapefiles (`polygon_shapefile`) as an input and provides landslide polygon images as an output to the function. It creates $N \times N$ ($N = 64$ in our case) pixel image with binary values of 0 or 255 for each pixel. This function first increase the number of data points in polygons using `increase_resolution_polygon` function and then takes a bounding box of polygon and transforms the coordinates

505 of polygons by subtracting polygon $(minimum\_easting, minimum\_northing)$ value from each point in the polygon. Then divide each point in polygon $(easting, northing)$ value by resolution of pixels (desired spatial distance between any two adjacent horizontally or vertically pixels) and convert them into nearest integers. Then for each pixel (x, y) the value of the pixel is 255 if there exists a point in the polygon with coordinates (x, y) otherwise the value of the pixel is 0. This function also removes those landslides having length and width of bounding box greater than 180 meters as the image of a polygon has some

510 restrictions on maximum landslide polygon it can have (resolution of pixels (3 meters) $\times$ N = 192 meters).

`train_augment (train_data, train_label)`: This function takes input training landslides data (`train_data`) and training labels (`train_label`) as input. The main idea behind using `train_augment` function is to augment the training data as CNN is data extensive algorithm. It rotates each image by 90°, 180°, 270°and flip the image vertically and

515 horizontally to increase the number of training samples. The output of the function is augmented training data and labels.

`classify_inventory_cnn (earthquake_inventory_images, rainfall_inventory_images, test_inventory_images)`: This function takes training earthquake inventory images (`earthquake_inventory_images`), training rainfall inventory images (`rainfall_inventory_images`) and testing inventory images (`test_inventory_images`)

520 as input to the function. Within the function, it combines an equal number of training earthquake and rainfall samples to avoid any class imbalance problem and then augments the training data by using the `train_augment` function. Then it trains the CNN algorithm on augmented training data and predicts the probability of testing landslides belonging to each of the trigger classes.

525 *Code availability.* The source code and future updates are available in the GitHub repository (`https://github.com/kamalrana7843/Landsifier.git`).

Data availability. The landslide inventories used in this paper are publicly available by Geospatial Information Authority (GSI) and the National Research Institute for Earth Science and Disaster Resilience (NIED). The 30 meters SRTM DEM data used is also publicly available by NASA and downloadable via (`https://www2.jpl.nasa.gov/srtm/`).

530 *Author contributions.* All authors contributed to the writing and reviewing of the manuscript. KR developed the code. NM and UO interpreted the results and supervised the work.

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

Acknowledgements. This project is supported by Co-PREPARE project (No: 57553291) by German Academic Exchange Service (DAAD). KR acknowledges support from RIT's Steven M. Wear Endowed graduate fellowship and NM acknowledges support through RIT's FEAD
535 grant. UO acknowledges funding from the Research Focus Point "Earth and Environmental Systems" of the University of Potsdam.

References

Adams, H., Emerson, T., Kirby, M., Neville, R., Peterson, C., Shipman, P., Chepushtanova, S., Hanson, E., Motta, F., and Ziegelmeier, L.: Persistence images: A stable vector representation of persistent homology, Journal of Machine Learning Research, 18, 2017.

Albawi, S., Mohammed, T. A., and Al-Zawi, S.: Understanding of a convolutional neural network, in: 2017 International Conference on Engineering and Technology (ICET), pp. 1–6, Ieee, 2017.

Amato, G., Palombi, L., and Raimondi, V.: Data–driven classification of landslide types at a national scale by using Artificial Neural Networks, International Journal of Applied Earth Observation and Geoinformation, 104, 102 549, 2021.

Arabameri, A., Chandra Pal, S., Rezaie, F., Chakrabortty, R., Saha, A., Blaschke, T., Di Napoli, M., Ghorbanzadeh, O., and Thi Ngo, P. T.: Decision tree based ensemble machine learning approaches for landslide susceptibility mapping, Geocarto International, pp. 1–35, 2021.

Aurisano, A., Radovic, A., Rocco, D., Himmel, A., Messier, M., Niner, E., Pawloski, G., Psihas, F., Sousa, A., and Vahle, P.: A convolutional neural network neutrino event classifier, Journal of Instrumentation, 11, P09 001, 2016.

Azar, A. T., Elshazly, H. I., Hassanien, A. E., and Elkorany, A. M.: A random forest classifier for lymph diseases, Computer methods and programs in biomedicine, 113, 465–473, 2014.

Barnett, I., Malik, N., Kuijjer, M. L., Mucha, P. J., and Onnela, J.-P.: EndNote: Feature-based classification of networks, Network Science, 7, 438–444, 2019.

Behling, R., Roessner, S., Segl, K., Kleinschmit, B., and Kaufmann, H.: Robust automated image co-registration of optical multi-sensor time series data: Database generation for multi-temporal landslide detection, Remote Sensing, 3, 2572–2600, 2014.

Behling, R., Roessner, S., Golovko, D., and Kleinschmit, B.: Derivation of long-term spatiotemporal landslide activity—A multi-sensor time series approach, Remote Sensing of Environment, 186, 88–104, https://doi.org/10.1016/j.rse.2016.07.017, 2016.

Belgiu, M. and Drăguţ, L.: Random forest in remote sensing: A review of applications and future directions, ISPRS journal of photogrammetry and remote sensing, 114, 24–31, 2016.

Bhattacharya, B. and Solomatine, D. P.: Machine learning in soil classification, Neural networks, 19, 186–195, 2006.

Biau, G.: Analysis of a random forests model, The Journal of Machine Learning Research, 13, 1063–1095, 2012.

Biau, G. and Scornet, E.: A random forest guided tour, Test, 25, 197–227, 2016.

Bíl, M., Raška, P., Dolák, L., and Kubeček, J.: CHILDA–Czech Historical Landslide Database, Natural Hazards and Earth System Sciences, 21, 2581–2596, 2021.

Breiman, L.: Random forests, Machine learning, 45, 5–32, 2001.

Bubenik, P. and Dłotko, P.: A persistence landscapes toolbox for topological statistics, Journal of Symbolic Computation, 78, 91–114, 2017.

Carlsson, G.: Topology and data, Bulletin of the American Mathematical Society, 46, 255–308, 2009.

Chaudhary, A., Kolhe, S., and Kamal, R.: An improved random forest classifier for multi-class classification, Information Processing in Agriculture, 3, 215–222, 2016.

Cruden, M. D. and Varnes, J. D.: Landslide Types and Processes, in: Landslides: Investigation and Mitigation, no. 247 in Special Report(National Research Council (U.S.) Transportation Research Board), pp. 36–75, National Academy Press, Washington, D.C, 1996.

Depicker, A., Jacobs, L., Mboga, N., Smets, B., Van Rompaey, A., Lennert, M., Wolff, E., Kervyn, F., Michellier, C., Dewitte, O., et al.: Historical dynamics of landslide risk from population and forest-cover changes in the Kivu Rift, Nature Sustainability, 4, 965–974, 2021.

Domingos, P.: A few useful things to know about machine learning, Communications of the ACM, 55, 78–87, 2012.

Fan, X., van Westen, C. J., Korup, O., Gorum, T., Xu, Q., Dai, F., Huang, R., and Wang, G.: Transient water and sediment storage of the decaying landslide dams induced by the 2008 Wenchuan earthquake, China, Geomorphology, 171, 58–68, 2012.

Farr, T. G., Rosen, P. A., Caro, E., Crippen, R., Duren, R., Hensley, S., Kobrick, M., Paller, M., Rodriguez, E., Roth, L., et al.: The shuttle radar topography mission, Reviews of geophysics, 45, 2007.

Friedman, J., Hastie, T., and Tibshirani, R.: The elements of statistical learning, vol. 1, Springer series in statistics New York, 2001.

Garin, A. and Tauzin, G.: A topological" reading" lesson: Classification of MNIST using TDA, in: 2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA), pp. 1551–1556, IEEE, 2019.

Ghorbanzadeh, O., Blaschke, T., Gholamnia, K., Meena, S. R., Tiede, D., and Aryal, J.: Evaluation of different machine learning methods and deep-learning convolutional neural networks for landslide detection, Remote Sensing, 11, 196, 2019.

Gillies, S.: The Shapely user manual, 2013.

Gorum, T., Korup, O., van Westen, C. J., van der Meijde, M., Xu, C., and van der Meer, F. D.: Why so few? Landslides triggered by the 2002 Denali earthquake, Alaska, Quaternary Science Reviews, 95, 80–94, 2014.

Guo, T., Dong, J., Li, H., and Gao, Y.: Simple convolutional neural network on image classification, in: 2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA), pp. 721–724, IEEE, 2017.

Guzzetti, F., Mondini, A. C., Cardinali, M., Fiorucci, F., Santangelo, M., and Chang, K.-T.: Landslide inventory maps: New tools for an old problem, Earth-Science Reviews, 112, 42–66, 2012.

Hensel, F., Moor, M., and Rieck, B.: A Survey of Topological Machine Learning Methods, Frontiers in Artificial Intelligence, 4, 52, 2021.

Jones, J. N., Boulton, S. J., Bennett, G. L., Stokes, M., and Whitworth, M. R.: Temporal variations in landslide distributions following extreme events: Implications for landslide susceptibility modeling, Journal of Geophysical Research: Earth Surface, 126, e2021JF006 067, 2021.

Krizhevsky, A., Sutskever, I., and Hinton, G. E.: Imagenet classification with deep convolutional neural networks, Advances in neural information processing systems, 25, 2012.

Kuhn, M. and Johnson, K.: Applied predictive modeling, vol. 26, Springer, 2013.

Kuhn, M., Johnson, K., et al.: Applied predictive modeling, vol. 26, Springer, 2013.

Kursa, M. B.: Robustness of Random Forest-based gene selection methods, BMC bioinformatics, 15, 8, 2014.

Li, Q., Cai, W., Wang, X., Zhou, Y., Feng, D. D., and Chen, M.: Medical image classification with convolutional neural network, in: 2014 13th international conference on control automation robotics & vision (ICARCV), pp. 844–848, IEEE, 2014.

Liaw, A., Wiener, M., et al.: Classification and regression by randomForest, R news, 2, 18–22, 2002.

Loche, M., Alvioli, M., Marchesini, I., Bakka, H., and Lombardo, L.: Landslide susceptibility maps of Italy: Lesson learnt from dealing with multiple landslide types and the uneven spatial distribution of the national inventory, Earth-Science Reviews, p. 104125, 2022.

Lombardo, L. and Tanyas, H.: From scenario-based seismic hazard to scenario-based landslide hazard: fast-forwarding to the future via statistical simulations, Stochastic Environmental Research and Risk Assessment, pp. 1–14, 2021.

Lombardo, L., Bakka, H., Tanyas, H., van Westen, C., Mai, P. M., and Huser, R.: Geostatistical modeling to capture seismic-shaking patterns from earthquake-induced landslides, Journal of Geophysical Research: Earth Surface, 124, 1958–1980, 2019.

Lombardo, L., Opitz, T., Ardizzone, F., Guzzetti, F., and Huser, R.: Space-time landslide predictive modelling, Earth-science reviews, 209, 103 318, 2020.

Lombardo, L., Tanyas, H., Huser, R., Guzzetti, F., and Castro-Camilo, D.: Landslide size matters: A new data-driven, spatial prototype, Engineering geology, 293, 106 288, 2021.

Malamud, B. D., Turcotte, D. L., Guzzetti, F., and Reichenbach, P.: Landslide inventories and their statistical properties, Earth Surface Processes and Landforms, 29, 687–711, 2004.

Marin, R. J., García, E. F., and Aristizábal, E.: Effect of basin morphometric parameters on physically-based rainfall thresholds for shallow landslides, Engineering Geology, 278, 105 855, 2020.

Martha, T. R., Roy, P., Jain, N., Khanna, K., Mrinalni, K., Kumar, K. V., and Rao, P.: Geospatial landslide inventory of India—an insight into occurrence and exposure on a national scale, Landslides, 18, 2125–2141, 2021.

Moreno, M., Steger, S., Tanyas, H., and Lombardo, L.: Modeling the size of co-seismic landslides via data-driven models: the Kaikōura's example, https://doi.org/10.31223/x5vd1p, 2022.

Munch, E.: A user's guide to topological data analysis, Journal of Learning Analytics, 4, 47–61, 2017.

Oksanen, J. and Sarjakoski, T.: Error propagation of DEM-based surface derivatives, Computers & Geosciences, 31, 1015–1027, 2005.

Okun, O. and Priisalu, H.: Random forest for gene expression based cancer classification: overlooked issues, in: Iberian conference on pattern recognition and image analysis, pp. 483–490, Springer, 2007.

Ozturk, U., Pittore, M., Behling, R., Roessner, S., Andreani, L., and Korup, O.: How robust are landslide susceptibility estimates?, Landslides, https://doi.org/10.1007/s10346-020-01485-5, 2020.

Ozturk, U., Saito, H., Matsushi, Y., Crisologo, I., and Schwanghart, W.: Can global rainfall estimates (satellite and reanalysis) aid landslide hindcasting?, Landslides, https://doi.org/10.1007/s10346-021-01689-3, 2021.

Pal, M.: Random forest classifier for remote sensing classification, International journal of remote sensing, 26, 217–222, 2005.

Qi, Y.: Random forest for bioinformatics, in: Ensemble machine learning, pp. 307–323, Springer, 2012.

Rana, K., Ozturk, U., and Malik, N.: Landslide Geometry Reveals its Trigger, Geophysical Research Letters, 48, e2020GL090 848, 2021.

Reininghaus, J., Huber, S., Bauer, U., and Kwitt, R.: A stable multi-scale kernel for topological machine learning, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4741–4748, 2015.

Rodriguez-Galiano, V. F., Ghimire, B., Rogan, J., Chica-Olmo, M., and Rigol-Sanchez, J. P.: An assessment of the effectiveness of a random forest classifier for land-cover classification, ISPRS Journal of Photogrammetry and Remote Sensing, 67, 93–104, 2012.

Saito, H., Uchiyama, S., Hayakawa, Y. S., and Obanawa, H.: Landslides triggered by an earthquake and heavy rainfalls at Aso volcano, Japan, detected by UAS and SfM-MVS photogrammetry, Progress in Earth and Planetary Science, 5, 15, https://doi.org/10.1186/s40645-018-0169-6, 2018.

Samia, J., Temme, A., Bregt, A., Wallinga, J., Guzzetti, F., Ardizzone, F., and Rossi, M.: Do landslides follow landslides? Insights in path dependency from a multi-temporal landslide inventory, Landslides, 14, 547–558, 2017.

Schmitt, R., Tanyas, H., Jessee, N., M.A., Z., Jing, Biegel, K.M., A., K.E., J., R.W., T., E.M., v. W., C.J., S., H.P., W., D.J., G., J.W., G., Tolga, X., Chong, R. E., and K.L., K.: An Open Repository of Earthquake-Triggered Ground-Failure Inventories (ver 2.0, December 2018), U.S. Geological Survey data release collection, https://doi.org/10.5066/F7H70DB4, 2018.

Stumpf, A. and Kerle, N.: Object-oriented mapping of landslides using Random Forests, Remote sensing of environment, 115, 2564–2577, 2011.

Tanyaş, H., Görüm, T., Kirschbaum, D., and Lombardo, L.: Could road constructions be more hazardous than an earthquake in terms of mass movement?, Natural hazards, 112, 639–663, 2022.

Tauzin, G., Lupo, U., Tunstall, L., Pérez, J. B., Caorsi, M., Medina-Mardones, A. M., Dassatti, A., and Hess, K.: giotto-tda:: A Topological Data Analysis Toolkit for Machine Learning and Data Exploration., J. Mach. Learn. Res., 22, 39–1, 2021.

Taylor, F., Malamud, B., Witt, A., and Guzzetti, F.: Landslide shape, ellipticity and length?to?width ratios, Earth Surface Processes and Landforms, 43, 3164–3189, 2018.

Varnes, D. J.: Slope movement types and processes, Special report, 176, 11–33, 1978.

Yamashita, R., Nishio, M., Do, R. K. G., and Togashi, K.: Convolutional neural networks: an overview and application in radiology, Insights into imaging, 9, 611–629, 2018.

Zhang, C. and Ma, Y.: Ensemble machine learning: methods and applications, Springer, 2012.

650