6-29-2021

Evaluation of deep learning against conventional limit equilibrium methods for slope stability analysis

Behnam Azmoon
Michigan Technological University, bazmoon@mtu.edu

Aynaz Biniyaz
Michigan Technological University, abiniyaz@mtu.edu

Zhen (Leo) Liu
Michigan Technological University, zhenl@mtu.edu

Recommended Citation

Azmoon, B., Biniyaz, A., & Liu, Z. (2021). Evaluation of deep learning against conventional limit equilibrium methods for slope stability analysis. *Applied Sciences (Switzerland), 11*(13). http://doi.org/10.3390/app11136060

Retrieved from: https://digitalcommons.mtu.edu/michigantech-p/15107

Follow this and additional works at: https://digitalcommons.mtu.edu/michigantech-p

 Part of the Civil and Environmental Engineering Commons

applied sciences

MDPI

Article

Evaluation of Deep Learning against Conventional Limit Equilibrium Methods for Slope Stability Analysis

Behnam Azmoon, Aynaz Biniyaz and Zhen (Leo) Liu *

Department of Civil and Environmental Engineering, Michigan Technological University, Houghton, MI 49931, USA; bazmoon@mtu.edu (B.A.); abiniyaz@mtu.edu (A.B.)
* Correspondence: zhenl@mtu.edu; Tel.: +1-906-487-1826

Featured Application: This paper presents a comparison study with well-controlled data to evaluate two new deep learning methods and their relationships and differences with traditional methods. We implemented four widely accepted limit equilibrium analysis methods and compared their implementations and results with the newly proposed deep learning methods. This will lend engineers a clear reference regarding how deep learning works in comparison with traditional methods. With this paper, readers can easily see the potential and technical advantages of the new methods. This presents a good example to show the comparison between traditional physics-based approaches and the data-driven approaches and demonstrate how data-driven approaches can change or complement the traditional engineering practices. The work will help bridge the gap between traditional engineering analysis of geosystems and advanced engineering informatics and explore "big data" solutions for many similar engineering applications (e.g., with mechanical or stability analysis).

Abstract: This paper presents a comparison study between methods of deep learning as a new category of slope stability analysis, built upon the recent advances in artificial intelligence and conventional limit equilibrium analysis methods. For this purpose, computer code was developed to calculate the factor of safety (FS) using four limit equilibrium methods: Bishop's simplified method, the Fellenius method, Janbu's simplified method, and Janbu's corrected method. The code was verified against Slide2 in RocScience. Subsequently, the average *FS* values were used to approximate the "true" *FS* of the slopes for labeling the images for deep learning. Using this code, a comprehensive dataset of slope images with wide ranges of geometries and soil properties was created. The average *FS* values were used to label the images for implementing two deep learning models: a multiclass classification and a regression model. After training, the deep learning models were used to predict the *FS* of an independent set of slope images. Finally, the performance of the models was compared to that of the conventional methods. This study found that deep learning methods can reach accuracies as high as 99.71% while improving computational efficiency by more than 18 times compared with conventional methods.

Keywords: convolutional neural networks; slope stability; multiclass classification; regression; deep learning; landslides; limit equilibrium methods

Citation: Azmoon, B.; Biniyaz, A.; Liu, Z. Evaluation of Deep Learning against Conventional Limit Equilibrium Methods for Slope Stability Analysis. *Appl. Sci.* **2021**, *11*, 6060. https://doi.org/10.3390/app11136060

Academic Editor: Luis Javier Garcia Villalba

Received: 1 June 2021
Accepted: 23 June 2021
Published: 29 June 2021

1. Introduction

Slope stability analysis is critical to the prevention and hazard mitigation of landslides. Especially in the present day, urbanization and population growth have been necessitating the build-up of terraces and corridors to make room for buildings and infrastructures, leading to more slope stability considerations in the built environment [1,2]. This raises the demand for the understanding, analysis, and prevention of landslides. The most common force-based methods for slope stability analysis are limit equilibrium methods (LEMs) [3], strength reduction methods (SRMs) [4], and limit analysis methods (LAMs).

LEMs were explored extensively in the early days to study slopes with hand-performed computations due to the lack of computing power. Despite the long history, these methods evolved with the availability of computers as well. Computers enabled LEMs to consider the internal forces, pore pressure, and multiple layers of soils. In addition, the simplicity of the underlying theories contributed to the popularity and widespread use of these methods among engineers [5,6]. However, LEMs are statically indeterminate problems, and the use of these methods requires assumptions of the internal forces that compromise their accuracy [7].

SRMs are the second category of methods that are commonly applied through the finite element, finite difference, and discrete element methods. These methods provide approximations to the exact solution of the governing equations of slope mechanics; therefore, they are more complicated than LEMs. Compared with LEMs, SRMs are advantageous in that they can consider strains [8]. Nonetheless, SRMs are relatively time-consuming, and their accuracy heavily relies on the accuracy of the considered geotechnical parameters [9].

Another group of approaches is LAMs, which make use of lower-bound and upper-bound theorems of plasticity [10]. While LEMs consider force and moment equilibrium along a specified slip surface, LAMs are rigorous. The reason for this is that, for the lower-bound, the stress field is in equilibrium with imposed loads at boundaries, while for the upper-bound, the velocity field solution is compatible with the imposed velocities. Despite the rigorousness, LAMs are not as popular as LEMs and SRMs due to the difficulties in constructing proper stress and velocity fields and obtaining optimal solutions. Additionally, the inclusion of pore water pressures, inhomogeneous soil profiles, and irregular slope geometries increase the complexity of LAMs. This can further complicate the manual construction of the stress and velocity fields, making LAMs impractical in most cases [11].

A more recent approach to slope stability is displacement-based analysis. This group of methods is focused on simulating the large movements and the post-failure behavior of slopes [12]. In many cases, the catastrophic damage caused by the deformations due to a landslide is more crucial than calculating the *FS* [13]. One of the most common methods in displacement-based analysis is the material point method (MPM) [14]. This method provides information for the internal development of shearing surfaces and the post-failure runout process [15]. The MPM has been used in numerous research projects and case studies. For example, Fern, et al. [16] used centrifuge tests to assess the effect of subsoil stiffness on the failure mechanism of dykes. Conte, et al. [17] utilized the MPM to investigate the runout process of the Maierato landslide.

Despite the clear underlying theories, the above categories of physics-based methods require assumptions to deal with the spatial variability of earth materials and their complex geotechnical behavior [18]. Moreover, the accurate implementation of these models to represent real-world conditions is time-consuming, computationally expensive, and requires a high level of expertise. Consequently, these models are either complex and impractical or over-simplified and inaccurate for field engineers when analyzing real-world problems. Additionally, these conventional methods are unable to take advantage of the large volumes of available data ("big data"), especially image and video data, and recent advancements in artificial intelligence.

Recently, a successor of artificial neural networks (ANNs), called convolutional neural networks (CNNs), gained popularity as a successful deep learning network. CNNs attracted an increasing amount of attention following their success in classifying 1.2 million high-resolution images into 1000 different categories [19]. These networks can significantly reduce model parameters and automatically extract data (image) features, and hence they generated remarkable improvements in learning outcomes [20].

Despite the success of CNNs in computer vision, they have only been explored in a few civil engineering applications [21]. Specifically, deep learning with CNNs has been used in structural health monitoring and vibration-based structural damage detection [22–26]. In geotechnical engineering, CNNs enabled researchers to use raw data and field records for assessing liquefaction potential [27], monitoring and predicting

landslide displacement [28,29], identifying the source location of microseismic events in underground mines [30], predicting the spatial correlation coefficients from cone penetration test data [31], and analyzing landslide susceptibility [32,33]. Despite this progress, the application of CNNs in classical geosystems such as slope stability analysis is still rare. As far as we know, the earliest and the only attempt is the authors' effort of using a CNN multiclass classifier for obtaining the *FS* of slopes, which was validated against Bishop's simplified method [34].

In this study, deep learning models are combined with physics-based models of slope stability to obtain new models for slope stability analysis. In particular, a multiclass classification model and a regression model were employed to automatically predict the *FS* of slope images. The performance of both models was evaluated against four widely adopted LEMs in terms of accuracy and computing efficiency: the ordinary method of slices (OMS), Bishop's simplified method (BSM), Janbu's simplified method (JSM), and Janbu's corrected method (JCM). The rest of the paper is organized as follows. Section 2 presents a brief introduction to the basics of deep learning with CNNs, followed by the theories of the proposed deep learning models. Next, Section 3 describes a procedure proposed for generating slope image data with different geometries and soil properties, as well as the labeling and pretreatment of the slope image data. The four LEMs employed in this study, including their theories and implementations, are detailed in Section 4. In Section 5, the results for training, validation, and testing are presented and analyzed, and conclusions are reached based on them.

2. Models

2.1. Overview of the Research Method

This subsection presents an overview of the research method and associated efforts. After going through the basic concepts and theories of deep learning with CNNs, the first step was to develop computer code for generating a dataset of artificial slope images with various geometries and soil properties. This code was then utilized to analyze the *FS*s of the images within the dataset using four LEMs, including the OMS, BSM, JSM, and JCM. The accuracy of the computer code was validated against Slide2 in RocScience. In the next step, the slope images were labeled and saved into lightning memory-mapped database (LMDB) format using two labeling procedures for the classification and regression models in deep learning. For the classification model, the slope images were grouped into nine classes based on the average values of their *FS*s. For the regression model, the images were labeled with the FS values. Considering that the LMDB format only supports integer values, the *FS*s of the slopes were multiplied by 1000, rounded down to the nearest integer and then used as the label. Next, the hyperparameters for the classification and regression models were defined in the solver. The solvers and datasets were then used to train the deep learning models. Subsequently, the trained models were employed to predict the *FS* values of an independent set of images to evaluate the performance of the newly developed methods. For this purpose, the *FS* predictions from the deep learning methods were compared against the *FS*s obtained by the previously mentioned LEMs regarding their accuracy and computing time. Finally, the results of these comparisons were analyzed, and conclusions were drawn based on them. In the following sections, more detailed descriptions will be provided for the essential components of the study.

2.2. Convolutional Neural Networks (CNNs)

CNNs were first proposed by LeCun [35], and they have been used frequently in computer vision applications ever since. CNNs usually consist of a series of convolutional layers, pooling layers, dropout layers, and fully connected layers. In convolutional layers, the model applies kernels to the image to detect features and generate various feature maps. The pooling layers introduce shift-invariance by reducing the resolution of the feature maps [36]. The dropout layers prevent overfitting, and batch normalization prevents internal covariate shift and speeds up training [37]. In the end, fully connected layers

produce the final output [38,39]. The Convolutional Architecture for Fast Feature Embedding (Caffe), developed by the Berkeley Vision and Learning Center (BVLC), was used as the deep learning framework in this study [40], considering that Caffe is a popular deep learning library that is fast, modular, and includes advanced deep learning techniques [41]. Two supervised learning models were proposed for conducting deep learning in this study: a multiclass classification model and a regression model. Both models were employed to study the slope image data to associate the features or images with their labels. In the following subsections, these two models are briefly discussed to show how deep learning can be used to obtain the *FS* values of slopes.

2.2.1. Multiclass Classification

Image classification used to be a challenging task involving two stages: using feature descriptors to extract handcrafted features and then feeding them to a trainable network. The problem with this approach was that it was heavily dependent on the first stage, which was a complicated task [35]. However, the availability of GPUs, better algorithms, and larger datasets helped address this problem and fueled the popularity of CNNs [42].

The main obstacle in using a multiclass classification model in this study was that the measure of the stability of slopes (i.e., the *FS*) was a continuous variable, while the classification model was inherently suitable for discrete variables. To address this issue, the *FS* values were grouped into nine categories based on the ranges of their *FS* values. The categories of *FS* values in Table 1. were adopted so that the classification model could reach accuracy to one decimal place. Aside from that, each range was associated with a unique label that was utilized in the multiclass class. The details of calculating the *FS* values of these slope images are discussed in the following sections.

Table 1. Categories of FS values and their associated labels for multiclass classification.

Category	Range of *FS*	Label
First	Less than 0.8	0
Second	0.8–0.9	1
Third	0.9–1.0	2
Fourth	1.0–1.1	3
Fifth	1.1–1.2	4
Sixth	1.2–1.3	5
Seventh	1.3–1.4	6
Eighth	1.4–1.5	7
Ninth	Greater than 1.5	8

To obtain a general understanding, let us assume there are k predefined classes in the data. Then, the goal of the multiclass classification model is to determine the input, x, belongs to which of the k categories. To achieve this goal, the deep learning model will approach a function f, which can be defined by its weights w to estimate the output y such that $y = f(x|w)$. In this study, the input fed to the network was an image or images in the form of an array or arrays of numbers representing pixel values of the image or images. The output was a vector of k numbers, in which all elements were equal to zero except the one that was equal to the label. Before calculating the output, the Softmax layer computed the probability of the input image belonging to every class. The Softmax layer takes the output of the last linear layer z_i and transforms it into the probabilities of the image belonging to each category by taking the exponents of each input and normalizing them over the sum of these probabilities:

$$p_i = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}} \quad (1)$$

where p_i is the probability that an image belongs to the ith category.

In deep learning, the goal is to minimize the error or loss. The loss function is a measure of how well the model fits the data. Cross-entropy is usually used to construct the loss function for the multiclass classification:

$$J(w) = -\sum_{i=1}^{k} [y_i \, ln(p_i)]. \quad (2)$$

Deep learning involves two stages: training and testing. In both stages, predictions are the goal, namely by using a CNN to predict the *FS* of a slope based on its image data. Before predictions, these CNNs must be trained on a dataset of labeled slope images. In training, the deep learning model is improved to minimize the loss function in the optimization process to find the best weights w for the dataset.

The CaffeNet architecture developed by the Berkeley Vision group was adopted for the classification model of this study [19]. Figure 1 illustrates the architecture of this model. This classifier is a one-GPU reproduction of AlexNet with minor improvements, including the removal of data augmentation and moving of the pooling layer in front of the normalization layer. Data augmentation is a technique that increases the amount of available data by synthetically modifying the existing data through procedures such as cropping, padding, and flipping. CaffeNet is more computationally efficient due to the size reduction obtained from the pooling layer [43].

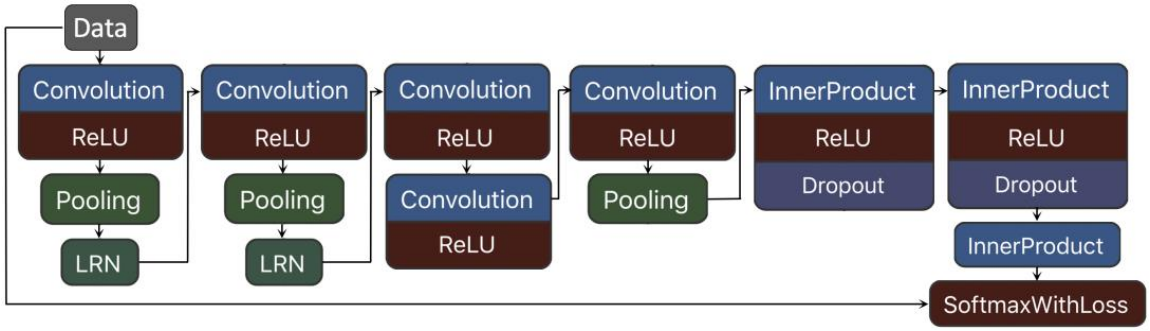


Figure 1. Architecture of the classification model utilized in this study.

2.2.2. Regression

Aside from classification, deep learning models are also employed to solve regression problems. Regression models are widely adopted for problems that involve the prediction of continuous values. Classification and regression models are quite similar, except for the format of their output variables and their loss layers. While the output variable in classification is a discrete value (categories), the output of regression can take any value in a continuous domain [44]. Moreover, the Softmax layer is commonly replaced with a fully connected regression layer with linear or a Sigmoid activation [45].

In regression, the goal is to train a network that can take the input x and predict the value of y as the output. In regression models, the network is trained on a labeled dataset and is required to produce a function $f : R^n \rightarrow R$. The procedure of training a regression model is quite similar to that of a classification model in that both of them aim to minimize the loss. As a result, the architecture of the regression model was almost identical to that of the classification model shown in Figure 1, except that the Softmax loss layer was replaced by a Euclidean loss layer. The Euclidean distance measures the distance between two real-valued vectors. Once this loss layer takes an InnerProduct layer as the input, it forms a linear least squares regression problem. The computed Euclidean loss was defined as

$$E = \frac{1}{2n} \sum_{n=1}^{N} \| \hat{y}_n - y_n \|_2^2 \quad (3)$$

where y_n is the true FS and $\hat{y}_n$ is the prediction of the FS for the input data [46]. The loss decreases when the Euclidean distance between the predictions and targets decreases, enabling the predictions made with the regression model to approach the true FS values of the slope images.

3. Data

The computer code was developed to produce a set of slope image data for the intended comparison study. The slope images within the dataset were different from each other in two aspects: the geometry and soil properties. In the following subsections, first, the process of producing different geometries will be concisely explained. Second, the procedure of incorporating the soil properties into the slope images will be described. Third, the validation procedure of the developed computer code will be presented. Finally, data pretreatment and labeling will be discussed.

3.1. Geometry

The application of a CNN in slope stability analysis is still in the preliminary stages. Therefore, this study adopted simple geometries to focus on understanding the concept and basics of such deep learning-based methods for slope stability analysis. As is shown in Figure 2, the slope images were produced on a 50 × 50 canvas. To account for different geometries, the x and y coordinates of the four specified points in this figure were produced using random numbers.

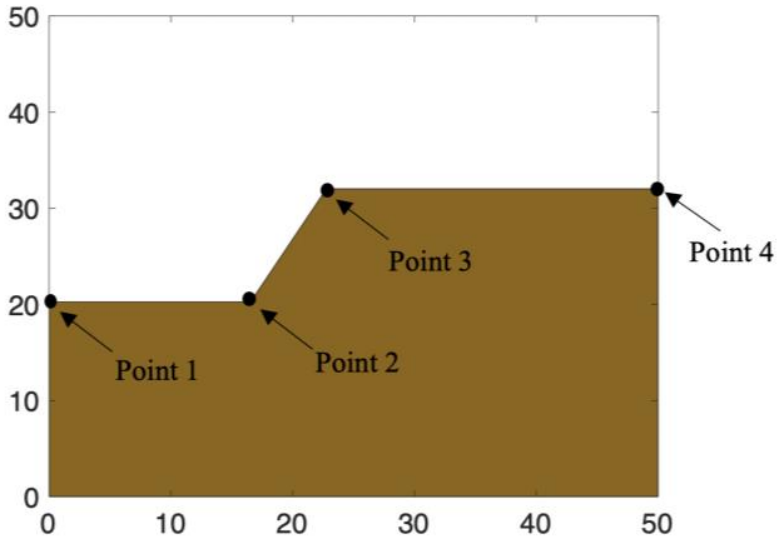


Figure 2. An example of the slope image data used in this study.

The equations for producing the coordinates of these four points are listed in Table 2, where the random variable λ is a uniformly distributed value between 0 and 1. In addition to the equation, some restrictions were adopted to obtain slope geometries that were significantly different and could be conveniently analyzed with LEMs. First, the y coordinates of Point 1 and Point 2 were the same, meaning that the bottom of the slope was always horizontal. Second, the crown of the slope was also horizontal. Third, the slip circles could not extend beyond the dimensions of the image data (50 × 50). Fourth, the slip circles with radii of less than one were neglected, since they were susceptible to shallow failure. Fifth, in the calculation of the FS, it was assumed that the center of the slip circle (x_c, y_c) had the following attributes: $0 < x_c < x_3$ and $y_3 < y_c < y_3 + 15$. These boundaries were then divided to form 30 grids in each direction, resulting in 900 slip circle centers for each image. Next, for each of these centers, 30 potential radii were tested to find the slip circle that could yield the minimum FS.

Table 2. Coordinates of the four points used to create the geometry of the slope image data.

Parameters	Descriptions	Formulations
(x_1, y_1)	Coordinates of Point 1	$(0, 15 + 10\lambda)$
(x_2, y_2)	Coordinates of Point 2	$(15 + 10\lambda, y_1)$
(x_3, y_3)	Coordinates of Point 3	$(x_2 + \lambda(29 - x_2), y_2 + 8 + \lambda(35 - y_2))$
(x_4, y_4)	Coordinates of Point 4	$(50, y_3)$
N	Number of slices	40

3.2. Soil Properties

The stability of the slopes is also primarily determined by the soil properties. Therefore, the soil properties were also considered in both the deep learning methods and LEMs in the comparison study. For this purpose, the soil properties, including the cohesion, friction angle, and unit weight, were selected based on the typical values of these parameters to reflect their variations in the real world [47,48]. Cohesionless soils were not considered in this study due to the shape and depth of their slip surfaces. The critical slip surface in cohesionless soils is a shallow plane parallel to the surface of the slope. However, the LEMs adopted in this study employ a circular slip surface to search for the circle that yields the minimum *FS*. The ranges of the adopted values for these parameters are listed in Table 3. For each image, computer code generated three random numbers within the given ranges with a uniform distribution to cover a wide range of soils. To incorporate these properties in the slope images, all of these values were normalized to values between 0 and 1. In the real slope images, the material properties were related to the information embedded in the image data; pixel values in different channels indicated the properties of the soil. In this pioneering study, this relationship was represented using a simple procedure. Each of the soil properties was assigned to one of the RGB channels of the image: red (R), green (G), or blue (B). The obtained color was then used to paint the slope image. In this way, the soil properties were carried by the image data in addition to the geometry.

Table 3. Range of soil properties used in this study.

Soil Properties	Range	Unit	Color
Cohesion (c')	10–100	kPa	Red
Friction angle (φ')	15–35	Degrees	Blue
Unit weight (γ)	17–22	kN/m^3	Green

3.3. Validation of the Computer Code for Data Generation

New code was developed for the *FS* calculations in this study because existing computer programs cannot be easily coupled with deep learning or its associated data treatments for the purpose of comparing deep learning against LEMs. This newly developed code was validated against a widely adopted commercial LEM software package, RocScience, before its use. A trial dataset of five images per category was randomly chosen to evaluate the accuracy of the developed code. These slopes were then simulated using Slide2 in RocScience. Figure 3 compares the *FS* predictions obtained with RocScience and those obtained with the new computer code. In this figure, the lines represent the *FS* values obtained in this study, while the markers represent the *FS*s calculated with RocScience. As can be seen, the *FS* values calculated with this code almost coincided with the results obtained with RocScience for the four LEMs.

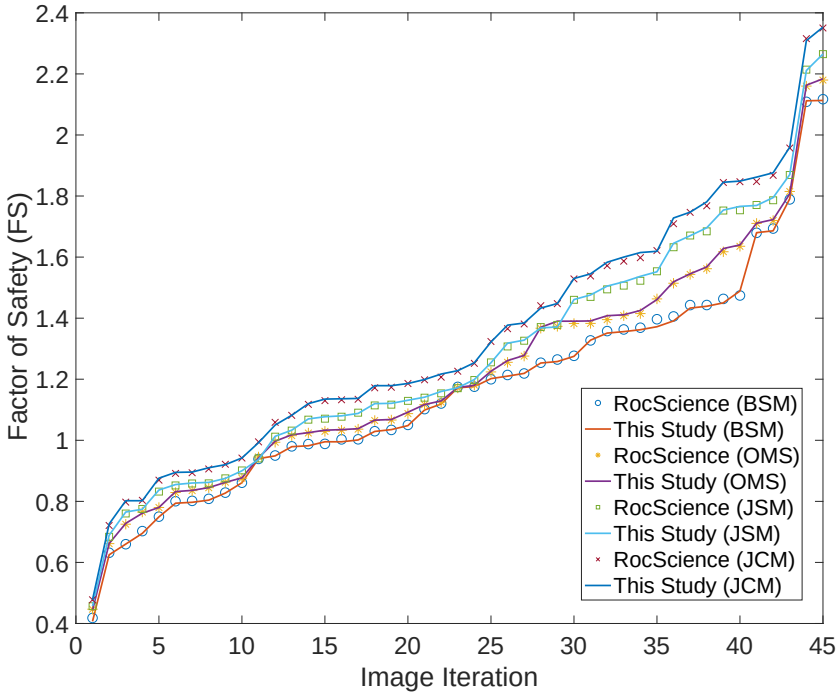


Figure 3. Comparison of *FS*s obtained with RocScience with the computer code.

3.4. Data Pretreatments

Several preprocessing techniques were applied to the slope images to improve the deep learning results [49,50]. In the first step, the histogram equalization technique was used to enhance the contrast of every image by decreasing the number of gray levels. $X = \{X(i,j)\}$ is used to represent an image, in which $X(i,j)$ is the gray level of the pixel. Given that n_k is the number of pixels with the X_k gray level, the probability density of X_k can be defined as

$$p(X_k) = \frac{n_k}{N},\ k = 0, 1, \ldots, L-1, \tag{4}$$

where N is the total number of pixels and the image density is digitized into L levels.

Then, the cumulative distribution can be formulated as

$$CDF(X_k) = \sum_{i=0}^{k} p(X_k). \tag{5}$$

Accordingly, the transform function of histogram equalization was calculated as

$$f(X_k) = X_0 + (X_{L-1} - X_0)CDF(X_k). \tag{6}$$

After histogram equalization, all slope images were resized to 227 × 227 pixels and then saved in lightning memory-mapped database (LMDB) format. Finally, the mean image was calculated and subtracted from each input image so that each feature has a similar range.

4. Limit Equilibrium Methods (LEMs)

LEMs were among the earliest methods in the analysis of slope stability. These methods are popular due to their simplicity and broad acceptance among practicing engineers [11]. There are multiple different LEMs, and the main differences between them are the various assumptions regarding the shape of the slip surface, interslice forces, and equilibrium considerations of the sliding mass [8,51]. Researchers have extensively investigated the accuracy of LEMs. The results of the LEMs were compared against methods with higher accuracy, such as finite element methods [52] with shear strength reduction

techniques [53,54]. The results from these comparisons indicated that the average *FS* calculated with LEMs was in good agreement with the more rigorous methods, especially for slopes with a homogenous soil layer.

However, there are some drawbacks inherent to the LEMs. One of these drawbacks is the assumption of a ductile stress–strain behavior due to the lack of information about the magnitude and variations of strains within the slope. In fact, it may not be correct to presume that the peak strength is mobilized simultaneously along the whole slip surface. Additionally, due to the fact that the number of equilibrium equations is less than the number of unknowns, LEMs are bound to use simplifying assumptions to render the problem determinate [7]. In the following subsections, considering this is a comparison study involving LEMs, the four LEMs selected in this study, together with their basic theories and advantages and disadvantages, are briefly discussed. The average of the *FS* values obtained by these four LEMs was adopted as the target value for labeling the image data. In theory, if LEMs can approximately calculate the *FS* in some way and the number of LEMs is big enough, we can assume that the average of the *FS* values calculated with the LEMs can approach the true *FS* of the slope in the image.

4.1. Bishop's Simplified Method (BSM)

This method considers the horizontal interslice forces and neglects the shear stresses between them [55]. The normal force acting at the center of the base of each slice is derived by adding the forces in the vertical direction. In other words, this method only ensures the force equilibrium in the vertical direction for each slice and derives the *FS* from the summation of moments about the center of the slip circle. It is important to note that because the resultant vector of the pore pressure and effective normal force passes through the center of the slip circle, these two forces do not affect the overall moment equilibrium. As a result, this method is not suitable for noncircular surfaces [56]. Although BSM does not consider the interslice shear stress, several studies have established that the *FS* was expected to differ by less than 5% from more rigorous methods [57,58]. Considering the equilibrium in the y direction, the *FS* in BSM is represented by the sum of the resisting forces divided by the sum of the driving forces:

$$FS = \frac{\sum\left\{(c'\Delta x + (w - u\Delta x)\tan\varphi')\frac{1}{M_\alpha}\right\}}{\sum w\sin\alpha}, \tag{7}$$

where Δx is the width of the slice, w is the weight of each slice, α is the angle between the potential failure arc and the horizontal forces at the midpoint of the slice, u is the pore pressure, and M_α is calculated as

$$M_\alpha = cos\alpha + \frac{\sin\alpha\tan\varphi'}{FS}. \tag{8}$$

4.2. Ordinary Method of Slices (OMS or Fellenius)

As one of the earliest LEMs, the OMS does not involve an iterative process for calculating the *FS*. This makes the OMS convenient for hand calculations. This method calculates the *FS* by considering the summation of moments about the slip surface's center. The major drawback of the OMS is that it assumes the resultant interslice forces are parallel to the base of the slice. This assumption results in the exclusion of the interslice forces and thus affects the accuracy of this method [57,59]. The equation for the *FS* in this method is

$$FS = \frac{\sum(c'l + N'\tan\varphi')}{\sum w\sin\alpha}, \tag{9}$$

where l is the slice base length and N' can be calculated as

$$N' = (w\cos\alpha - ul). \tag{10}$$

There are concerns regarding the accuracy of the OMS. Whitman and Bailey [60] claimed that this method could have errors of up to 60%. The accuracy of the OMS was investigated for the simple geometries and homogenous soils in this study. The OMS and BSM were used to calculate the *FS* values of 500 random slopes. Figure 4a shows a comparison between the *FS*s obtained by the OMS and BSM. This subplot illustrates good agreement between the two methods. In Figure 4b, the relative difference between the two methods is presented. The relative difference D_R is defined as

$$D_R = \frac{FS_{\mathrm{BSM}} - FS_{\mathrm{OMS}}}{FS_{\mathrm{BSM}}} \times 100\% \tag{11}$$

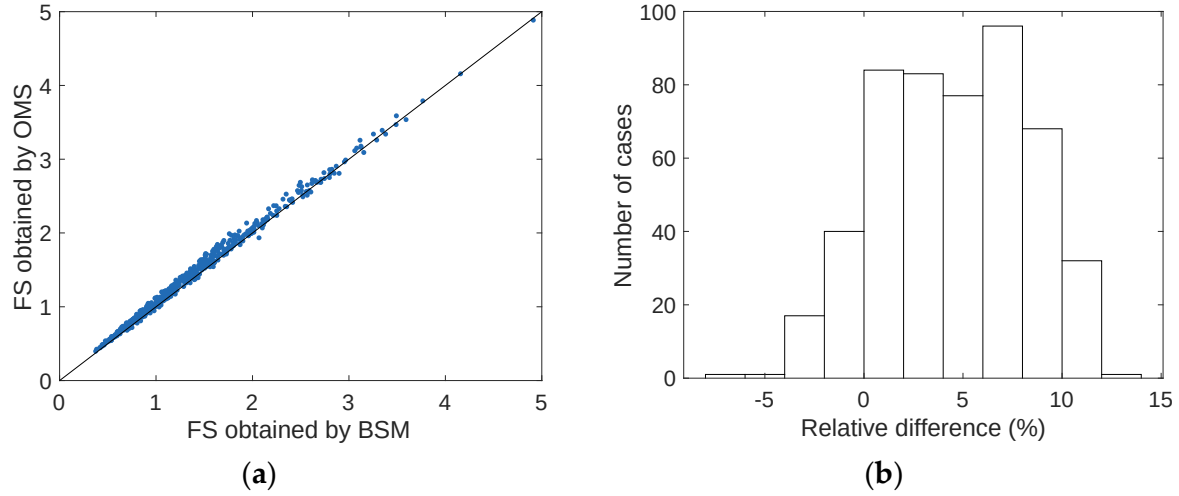


Figure 4. Evaluation of the performance of the OMS against BSM in (**a**) QQ plot of obtained *FS*s and (**b**) histogram of relative differences.

Figure 4b illustrates that, although the *FS* predictions of BSM were higher than the OMS in most cases, the discrepancy between the results of the two methods was not significant. The highest relative difference between the two methods was 12.19%, and the mean absolute error of the OMS with respect to the BSM was 0.0653. This comparison of *FS* results obtained with the OMS and BSM confirmed the appropriateness of using the OMS for such slopes.

4.3. Janbu's Simplified Method (JSM)

Janbu developed some of the most popular LEMs for slope stability analysis [61–63], from which Janbu's simplified and Janbu's corrected methods were employed in this study. While JSM was originally developed to handle composite slip surfaces, it is also capable of handling circular slip surfaces. Similar to BSM, JSM is an iterative method that considers the interslice normal forces and neglects the shear forces between the slices. JSM is a force equilibrium method, meaning that it satisfies the horizontal and vertical force equilibrium requirements but does not ensure moment equilibrium. This method calculates the *FS* as follows:

$$FS = \frac{\sum\left\{\left(\Delta x c' + (w-u)\tan\varphi'\right)\frac{1}{n_\alpha}\right\}}{\sum w \tan\alpha}, \tag{12}$$

where

$$n_\alpha = \cos^2\alpha\left(1 + \tan\alpha\frac{\tan\varphi'}{FS}\right). \tag{13}$$

4.4. Janbu's Corrected Method (JCM)

Janbu proposed a correction factor f_0 to consider the effect of interslice shear forces that were neglected in JSM. JCM uses the FS calculated via JSM (i.e., FS_{JSM}) and applies this correction factor to it to obtain a new FS (i.e., FS_{JCM}):

$$FS_{JCM} = FS_{JSM} \times f_0 \tag{14}$$

Janbu's correction factor is based on a series of curves for different soils. These curves were obtained by comparing the FS values of Janbu's generalized method [64] and JSM. The correction factor was later described based on these curves as

$$f_0 = 1.0 + b_1 \left[\frac{d}{L} - 1.4 \left(\frac{d}{L} \right)^2 \right], \tag{15}$$

where d and L are the depth and length of the slip surface, respectively (Figure 5), and b_1 is a parameter that depends on the soil type and can be obtained with Equation (16). It is worthwhile to mention that the correction factor is always greater than one. As a result, the FS calculated with JCM is 5–12% greater than that with JSM [57]:

$$b_1 = \begin{cases} for\ c\ only\ soils = 0.69 \\ for\ \varphi\ only\ soils = 0.31 \\ for\ c\ and\ \varphi\ soils = 0.5 \end{cases} \tag{16}$$

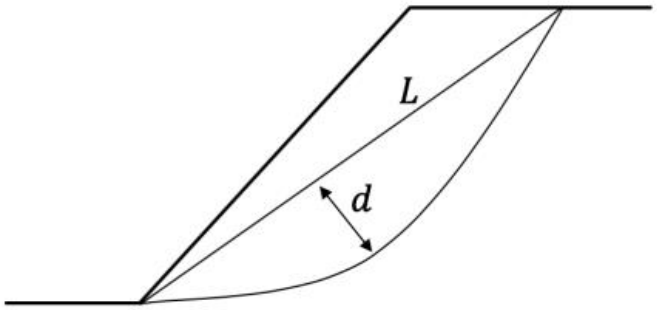


Figure 5. Depth and length of the slip surface.

5. Results and Discussions

5.1. Training

A dataset consisting of 37,500 slope images for training and 7500 for cross-validation was used to train the deep learning models. Transfer learning was adopted as the training method. In transfer learning, we used a CNN that was pretrained on a general large (source) dataset and repurposed the learned features for training on a target dataset. This improves learning in a new task through the transfer of knowledge from the previous task [65,66]. To achieve this, the BAIR Reference CaffeNet model that was trained on the ImageNet dataset with 1000 classes was utilized as the source dataset. Using the weights of this pretrained model as the starting point of our training helped speed up the training and significantly improved the performance of the models.

AdaDelta was adopted as the optimizer for the regression and classification models due to its data modalities, different model architecture choices, and robustness to noisy gradient information [67]. Additionally, both models used the same set of hyperparameters that had yielded the best results: a "fixed" learning rate policy with a base learning rate of 0.05 and a maximum iteration of 40,000. Seventy-five test iterations were performed at an interval of 500 iterations. Other adopted hyperparameters included a momentum of 0.9, a snapshot of 5000, and a weight decay of 0.0005. Figure 6 demonstrates the training process for the classification model with these parameters. In this figure, the downward trend of

the loss indicates that the model was learning from the data. The maximum accuracy and loss obtained by this model were 82% and 0.2, respectively. Additionally, the regression model achieved a Euclidean loss of 2175 at the end of the training.

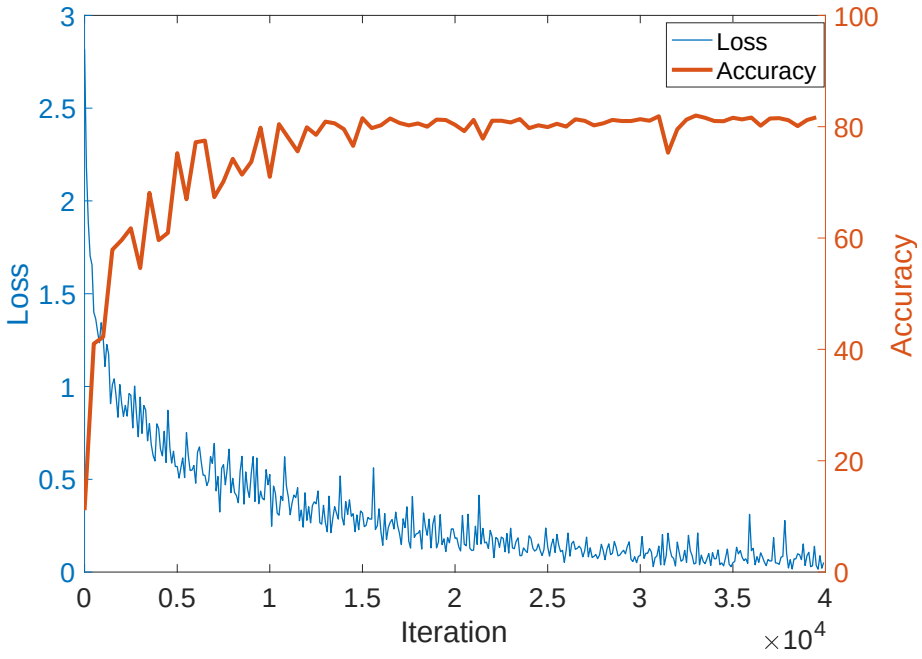


Figure 6. Learning curve for the classification model.

5.2. Testing

As was mentioned before, the goal of this study was to use both trained models to predict the *FS*s of slopes based on their images. To provide an unbiased assessment of the prediction performance of the models, a comprehensive testing dataset that was independent of the training images was needed. In addition, the testing dataset should have included cases from all ranges of the *FS*. Accordingly, more than 30,000 cases were generated using the computer code. Table 1 was then used to distribute these images into nine classes based on their *FS*s. One thousand slope images per category were randomly selected to form the testing dataset. The result was a dataset of 9000 slope images, for which the true *FS* and the range or class that it belonged to was known. The trained classification and regression models were then employed to predict the range and the value of the true *FS*, respectively. It is important to note that the inputs in the testing phase of the study were the unlabeled slope image data. Therefore, the deep learning models did not get any information regarding the true *FS*s of the slopes in the testing data. In the following, the prediction accuracy and computational efficiency of the deep learning-based methods and LEMs are compared against one another.

Figure 7 compares the predictions made with the trained models against the true *FS*s obtained with the computer code. The x-axes in both subplots represent the true *FS* values obtained with the computer code, and the y-axes are the *FS* values predicted with the deep learning models. In Figure 7a, both axes are divided into nine regions, corresponding to the number of classes in the classification model. In this subplot, each cell is color-coded based on the number of images that it represents. These numbers are also shown on each cell. The cells on the diagonal line indicate slopes in which the predicted and true *FS*s are identical. These cells are colored light green, which suggests a high number of images according to the colormap of this plot. The dark green cells represent images for which the distance between the predicted and true *FS* values was 0.1 (one category). The red cells represent *FS* predictions that were off by more than 0.1 (more than one category). The blank cells imply that there were no cases in these cells. Further analysis of Figure 7a shows that the testing

accuracy of the classification model was 80.9%, which was obtained by calculating the sum of the values on the diagonal line (7283) and dividing it by the total number of cases (9000). Additionally, the mean absolute error of prediction was 0.0195. The low gap between the training (82%) and testing accuracy (80.9%) indicates the absence of overfitting.

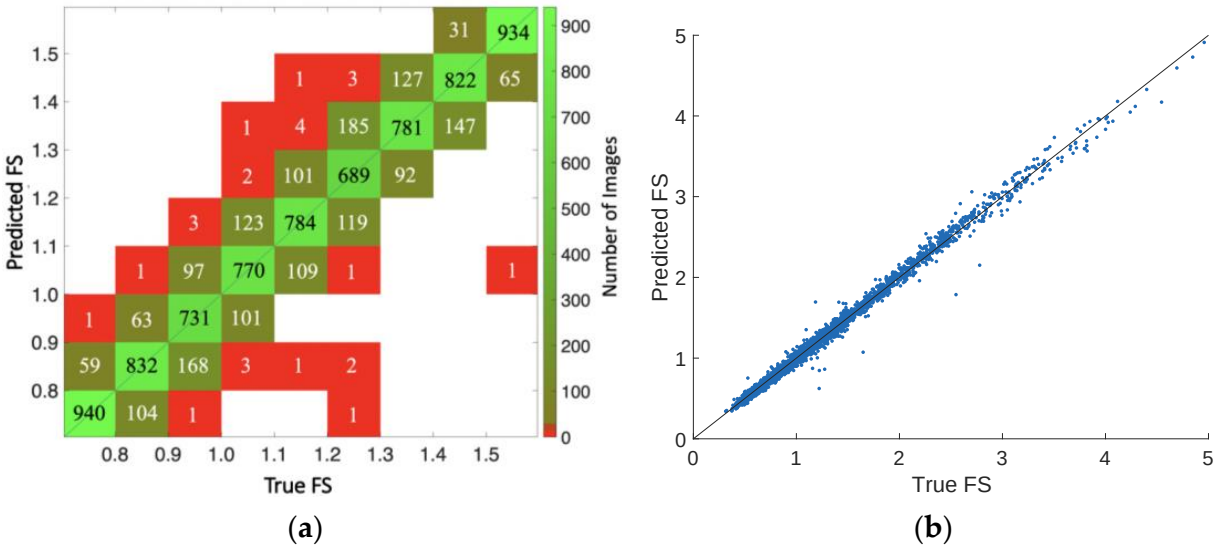


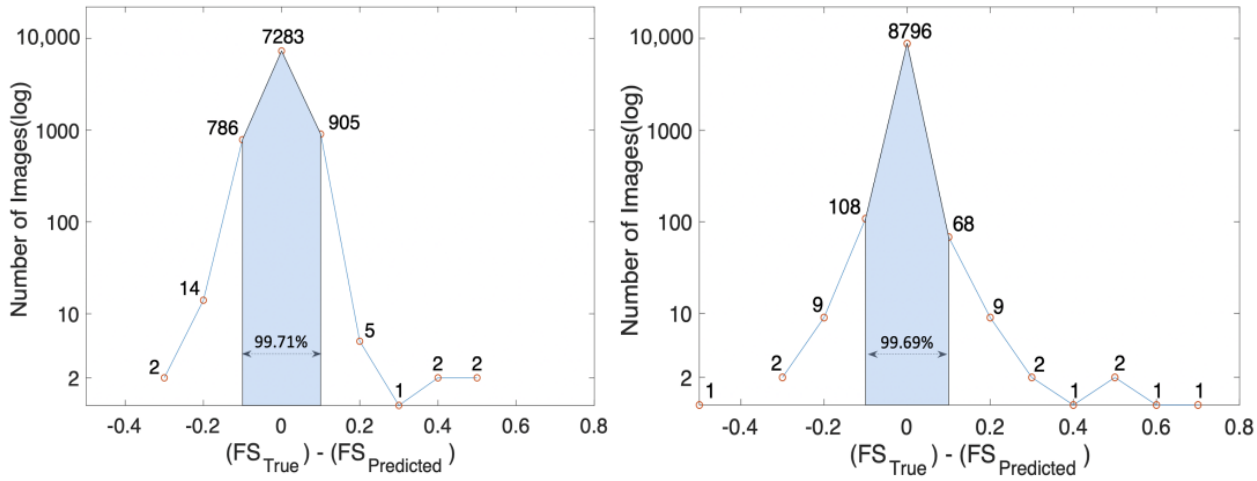
Figure 7. Results of testing for (**a**) the classification model and (**b**) the regression model.

In Figure 7b, each dot represents one image in the testing dataset. The prediction accuracy of the regression model can be interpreted as the distance between the dots in this plot and the diagonal line. Thus, the closer a dot is to the diagonal line ($y = x$), the more accurate its predicted *FS* value is. As can be seen in this plot, while in some cases the predicted *FS* was far from the true value, the number of such cases was small compared with the total number of cases (9000). The mean absolute error for the regression model was 0.0265.

The accuracy of classification was 80.9%, which does not appear to be high from a deep learning perspective. However, the accuracy could be much higher from the perspective of slope stability analysis. The reason for this is that any incorrect classification is counted as "bad" in deep learning. However, in slope stability analysis, the effectiveness of the results can also be measured by the distance between the predicted classification, (i.e., the *FS*) and the real classification; that is, predictions that are not far from the real classification (i.e., an *FS* value that is 0.1 above or below the true *FS*) is not "accurate" in deep learning but can still be acceptable in slope stability analysis. A reinterpretation of the results in Figure 7a revealed that more than 98% of the incorrectly predicted cases in deep learning would be acceptable in slope stability analysis.

The large number of cases that were one category off was potentially caused by the defined ranges. To gain more insights, we divided a continuous variable (*FS*) into nine categories to utilize a classification model. While this is a necessary step to prepare the dataset for the classification model, it may also lead to the loss of information. Once the labels are created, this loss of information can affect the classification model in the following ways. First, the model is unable to reflect the difference between the *FS* values of images that belong to the same category. Second, the model is incapable of distinguishing between images that have close *FS* values but belong to two nearby categories. We can reduce this loss of information by choosing smaller ranges, leading to a higher number of categories. The regression model here can be viewed as a classification model with an infinite number of categories or classes. The effect of this loss of information can be better quantified with statistical analysis. Figure 8 is presented to investigate the magnitude of errors in the predictions. In these semi-logarithmic plots, the *x*-axis denotes the error in

the predictions of the models, and the *y*-axis displays the number of slope images with a particular error. As is shown, if the predictions that were off by one category were considered acceptable, the accuracy of the regression and classification models would be 99.69% and 99.71%, respectively. Additionally, in the deep learning perspective, the regression model outperformed the classification model and decreased the number of predictions that were off by one category from 1691 to 176. This increased the deep learning accuracy from 80.92% in the classification model to 97.73% in the regression model. This leap in performance confirms that the loss of information associated with grouping *FS* values was responsible for a large portion of the errors in the classification model.



(**a**) (**b**)

Figure 8. Number of images versus error in *FS* predictions for (**a**) the classification model and (**b**) the regression model.

To further evaluate the effect of this loss of information, 500 random slope images were selected from the testing data. In Figure 9, the predicted range of *FS* values obtained by the classification model is compared with the true *FS* values. In this figure, the vertical axis is the true *FS* value of each point or image, and the horizontal axis is the predicted *FS* value. It is important to note that the classification model predicts the range of *FS*s, and the horizontal placement of points within a single range is for plotting purposes. The points that are located within the green boxes represent cases in which the predicted *FS* value was correct. By contrast, the points in red boxes belong to incorrectly predicted ranges. This figure shows that for the majority of the incorrectly predicted cases, the true *FS* value was quite close to the boundaries of the *FS* ranges defined for the classification model. This confirms that the loss of information made it difficult for the classification model to predict the range of *FS* values when the true *FS* was close to the cell boundaries.

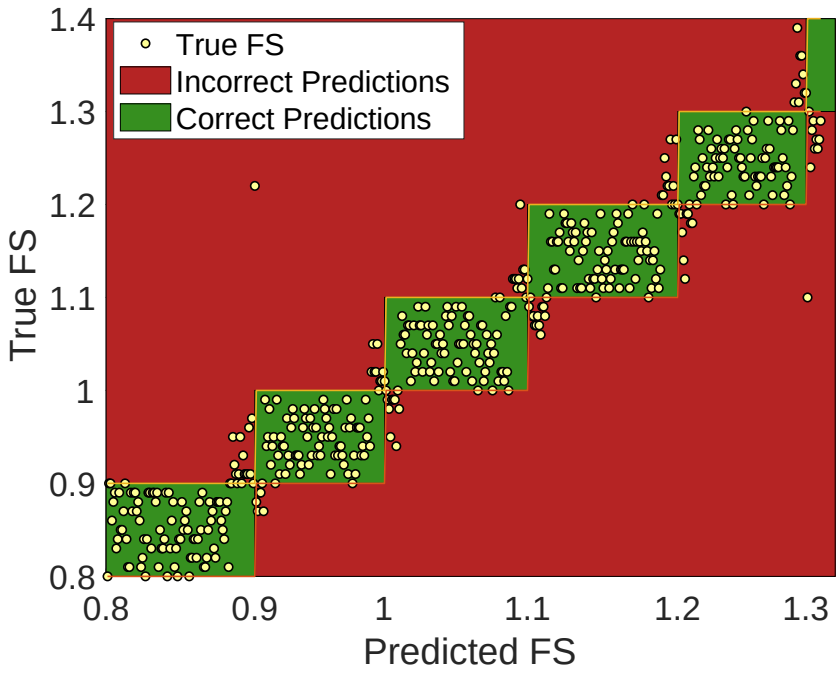


Figure 9. The comparison between the LEM and the classification model.

Figure 10 plots the *FS* values obtained by different analysis methods for a subset of the testing data. This figure contains 226 randomly selected slope images. The slopes were sorted based on their true *FS*s. A solid line was then used to represent the true *FS* values of the slopes. There are five types of markers in this figure: four of them belong to the four LEMs implemented in this study, and one is for the predictions of the regression model. The results are demonstrated in two sections for plotting purposes. The main plot shows 200 images for *FS* values up to 1.8, and the subplot at the upper left corner shows 26 slopes with higher *FS*s. The comparison indicates that all of the predicted *FS*s were quite close to the true *FS*s calculated as the average *FS* of the four LEMs. Further analysis of this plot shows that the BSM underestimated the *FS*s for most slopes, while JCM and JSM tended to overestimate them. Despite this difference, the predictions of the regression model, true *FS*s, and the OMS results were similar to one another. This plot also shows that, for the majority of the slopes, the predictions of the deep learning models were within the highest and lowest *FS*s obtained with the four LEMs and were more accurate than all of them. Another important observation from this figure is the ability of the regression model to distinguish between images within the first and ninth categories. While there was only one class for each of these wide ranges in the classification model, the regression model could estimate the *FS* values of these images instead of categorizing them as "lower than 0.8" (first category) or "higher than 1.5" (ninth category).

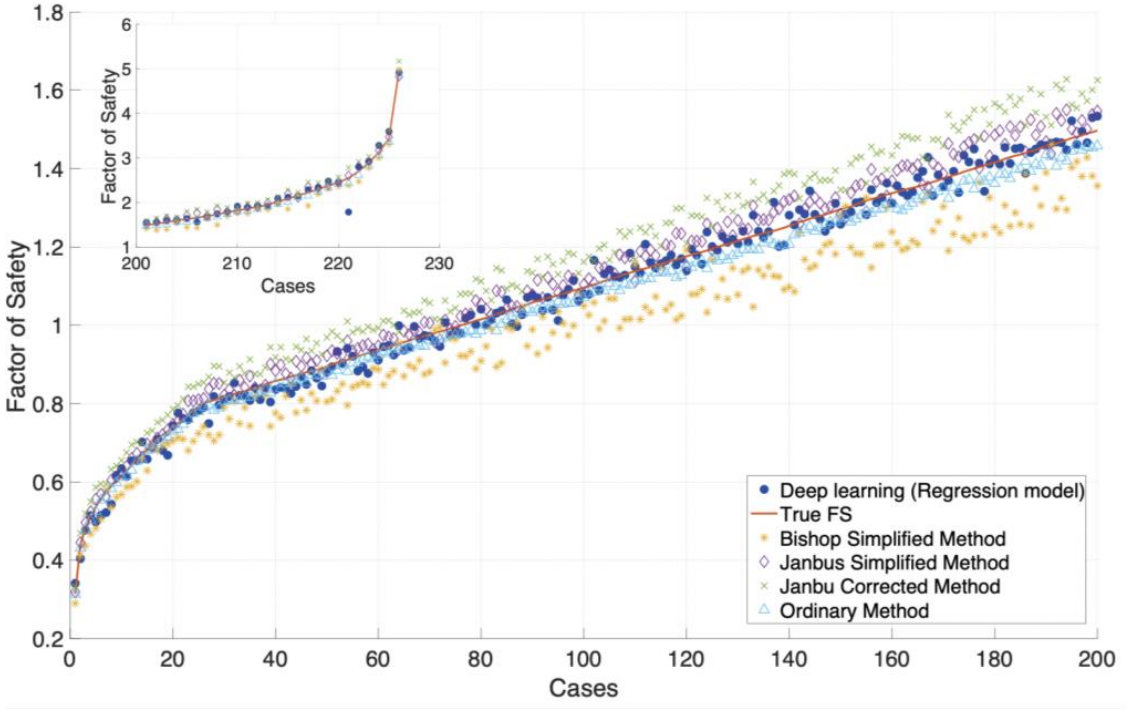


Figure 10. The comparison between LEMs and the regression model.

The other major criteria adopted in this comparison study were the computational demands and time-efficiency. Using LEMs to evaluate slope stability is a time-consuming process. This process is comprised of two main stages: manual construction of the model and using it to evaluate slope stability. By contrast, the ability of deep learning to quickly analyze large amounts of data is one of the main reasons for its popularity and widespread use. In this study, the amount of time needed for the LEMs to calculate the *FS* values (the second stage) was compared with that of the two proposed deep learning models. It is noted that the computing times for the classification and regression models were almost the same, so the computing time of the regression model is not shown in the plot. Figure 11 summarizes a comparison of the computing times needed for calculating the *FS* with the LEMs and that with the classification model. In this figure, each line with markers represents the deep learning model or a traditional LEM for calculating the *FS*. This figure shows that the deep learning model outperformed the traditional LEMs in terms of computation time. In addition, the amount of time needed for the LEMs increased approximately linearly as the number of cases increased. For example, the time needed to calculate the *FS*s of 200 slopes was almost twice the time needed for 100 slopes. However, deep learning methods behave differently. Every time a deep learning model is deployed, it first runs several background processes and then starts predicting *FS* values. Therefore, as the number of images increased, this initial cost was distributed over more images. That is why the deep learning model outperformed the LEMs more and more as the number of images in the testing dataset increased. To emphasize this boost in performance, the elapsed time for the classification model (blue line) and JCM (green line) were also labeled. For example, it took 15.1 s for the classification model to predict the *FS* for 200 slopes, while obtaining the *FS* via JCM for the same number of images took 1315 s. This means that deep learning methods are over 18 times more efficient than JCM. Aside from that, when analyzing 200 cases, the improved computational efficiency of the deep learning methods would be more notable if the time needed for manual procedures prior to running the model (the first stage) were considered for traditional LEMs. The traditional methods require a great deal of time to manually prepare the input, construct the model, and set up the geometry. However, deep learning methods can analyze raw image data and do not require any manual data preprocessing work once they are trained.

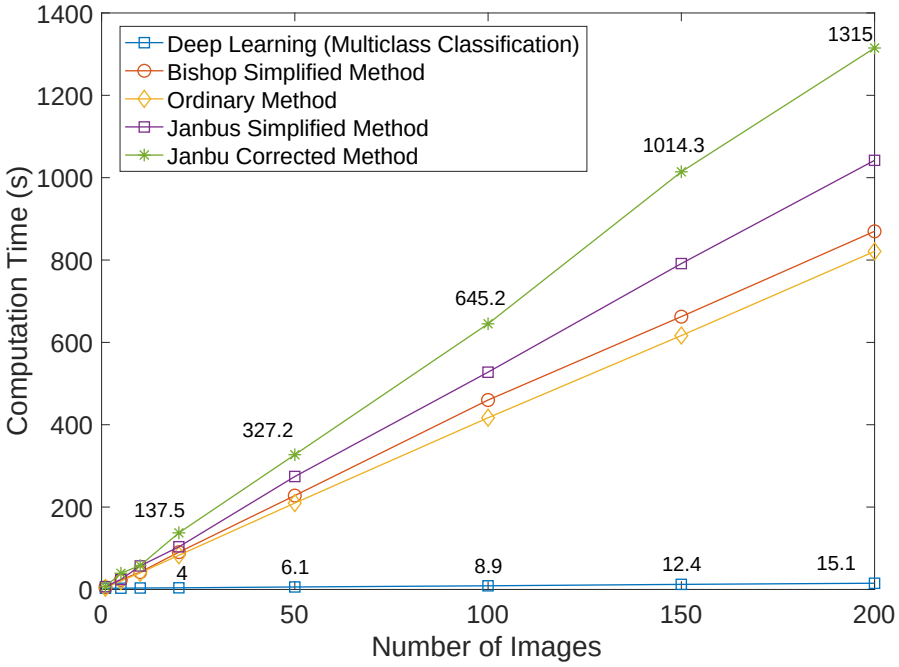


Figure 11. Computing time for the classification model and the LEM code.

6. Conclusions

This paper introduces deep learning-based methods as a new category of methods to evaluate slope stability. To achieve this, a multiclass classification model and a regression model in deep learning were employed to develop two slope stability analysis methods in this category. The performance of these methods was assessed in different aspects. The new methods exhibited promising performance in terms of both accuracy and computational efficiency. For accuracy, the classification model showed an accuracy of 82% in training and 80.9% for testing in a deep learning perspective. Further analysis revealed that, if the predictions that were off by one category were also considered acceptable, the testing accuracy would get as high as 99.71%. This indicated that the loss of information caused by using categories to represent a continuous variable (*FS*) was the reason for the lower accuracy of the classification model. By contrast, the use of the regression model prevented this loss of information and increased the deep learning accuracy to 97.73%. Aside from high accuracy, the ability of CNNs to analyze raw image data obviates the need for the time-consuming and error-prone procedures of constructing conventional models, which are required for LEMs and SRMs. Using raw data directly also means that we can avoid the simplifying assumptions associated with conventional methods, which could jeopardize the accuracy of stability analysis. Another advantage of using CNNs is their efficiency; much shorter computing times, compared with conventional methods, are needed with the same computing resources. The results showed that, even without considering the time needed for manual construction of the LEMs, the proposed CNNs were 18 times faster than JCM when analyzing 200 cases. It is also noteworthy that this difference in the computing time would even increase as the size of the data increased.

This study can pave the way toward adopting deep learning to analyze complex geosystem and geohazard problems and can be extended to other stability problems in engineering. It is also worthwhile to mention that the proposed models are based upon traditional LEMs and are limited by the constraints in their training data. Therefore, future studies should consider easing the constraints on simple geometries, using inhomogeneous soil properties, and incorporating pore pressure to make the study more practical. This concept can be further developed to use remote sensing and geographic information systems to analyze the stability of slopes in real time.

Author Contributions: Conceptualization and methodology, Z.L. and B.A.; software and validation, B.A. and A.B.; writing—original draft preparation, B.A.; writing—B.A. and Z.L.; supervision and project administration, Z.L. All authors have read and agreed to the published version of the manuscript.

Funding: The authors would like to acknowledge the financial support from the United States National Science Foundation (NSF) via Award 1742656 from the Geotechnical Engineering and Materials Program (now part of CMMI ECI). This work also benefited from the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by the National Science Foundation grant number ACI-1548562.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data will be available upon request.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Duncan, J.M.; Wright, S.G.; Brandon, T.L. *Soil Strength and Slope Stability*; John Wiley & Sons: Hoboken, NJ, USA, 2014.
2. Cheng, Y.M.; Lau, C. *Slope Stability Analysis and Stabilization: New Methods and Insight*; CRC Press: Boca Raton, FL, USA, 2014.
3. Zhu, D.; Lee, C.; Jiang, H. Generalised framework of limit equilibrium methods for slope stability analysis. *Geotechnique* **2003**, *53*, 377–395. [CrossRef]
4. Zienkiewicz, O.C.; Humpheson, C.; Lewis, R.W. Associated and non-associated visco-plasticity and plasticity in soil mechanics. *Géotechnique* **1975**, *25*, 671–689. [CrossRef]
5. Price, V.; Morgenstern, N. The analysis of the stability of general slip surfaces. *Geotechnique* **1965**, *15*, 79–93. [CrossRef]
6. Chen, R.H.; Chameau, J.-L. Three-dimensional limit equilibrium analysis of slopes. *Géotechnique* **1983**, *33*, 31–40. [CrossRef]
7. Duncan, J.M. State of the Art: Limit Equilibrium and Finite-Element Analysis of Slopes. *J. Geotech. Eng.* **1996**, *122*, 577–596. [CrossRef]
8. Pourkhosravani, A.; Kalantari, B. A review of current methods for slope stability evaluation. *Electron. J. Geotech. Eng.* **2011**, *16*, 1245–1254.
9. Feng, X.; Li, S.; Yuan, C.; Zeng, P.; Sun, Y. Prediction of Slope Stability using Naive Bayes Classifier. *KSCE J. Civ. Eng.* **2018**, *22*, 941–950. [CrossRef]
10. Yu, H.S.; Salgado, R.; Sloan, S.; Kim, J.M. Limit Analysis versus Limit Equilibrium for Slope Stability. *J. Geotech. Geoenviron. Eng.* **1998**, *124*, 1–11. [CrossRef]
11. Kim, J.; Salgado, R.; Lee, J. Stability Analysis of Complex Soil Slopes using Limit Analysis. *J. Geotech. Geoenviron. Eng.* **2002**, *128*, 546–557. [CrossRef]
12. Soga, K.; Alonso, E.; Yerro, A.; Kumar, K.; Bandara, S. Trends in large-deformation analysis of landslide mass movements with particular emphasis on the ma-terial point method. *Géotechnique* **2016**, *66*, 248–273. [CrossRef]
13. Troncone, A.; Pugliese, L.; Lamanna, G.; Conte, E. Prediction of rainfall-induced landslide movements in the presence of stabilizing piles. *Eng. Geol.* **2021**, *288*, 106143. [CrossRef]
14. Sulsky, D.; Chen, Z.; Schreyer, H. A particle method for history-dependent materials. *Comput. Methods Appl. Mech. Eng.* **1994**, *118*, 179–196. [CrossRef]
15. Alonso, E.E. Triggering and motion of landslides. *Géotechnique* **2020**, *71*, 1–57. [CrossRef]
16. Fern, E.J.; de Lange, D.A.; Zwanenburg, C.; Teunissen, J.A.M.; Rohe, A.; Soga, K. Experimental and numerical investigations of dyke failures involving soft materials. *Eng. Geol.* **2017**, *219*, 130–139. [CrossRef]
17. Conte, E.; Pugliese, L.; Troncone, A. Post-failure analysis of the Maierato landslide using the material point method. *Eng. Geol.* **2020**, *277*, 105788. [CrossRef]
18. Hsu, S.-C.; Nelson, P.P. Material Spatial Variability and Slope Stability for Weak Rock Masses. *J. Geotech. Geoenviron. Eng.* **2006**, *132*, 183–193. [CrossRef]
19. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
20. O'Shea, K.; Nash, R. An introduction to convolutional neural networks. *arXiv* **2015**, arXiv:1511.08458.
21. Aggarwal, C.C. *Neural Networks and Deep Learning*; Springer International Publishing: New York, NY, USA, 2018.
22. Abdeljaber, O.; Avci, O.; Kiranyaz, S.; Gabbouj, M.; Inman, D.J. Real-time vibra-tion-based structural damage detection using one-dimensional convolutional neural networks. *J. Sound Vib.* **2017**, *388*, 154–170. [CrossRef]
23. Zhang, L.; Yang, F.; Zhang, Y.D.; Zhu, Y.J. Road crack detection using deep convolutional neural network. In Proceedings of the 2016 IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, USA, 25–28 September 2016; pp. 3708–3712.
24. Oullette, R.; Browne, M.; Hirasawa, K. Genetic algorithm optimization of a con-volutional neural network for autonomous crack detection. In Proceedings of the 2004 Congress on Evolutionary Computation, Portland, OR, USA, 19–23 June 2004; pp. 516–521.

25. Zhang, A.; Wang, K.C.P.; Li, B.; Yang, E.; Dai, X.; Peng, Y.; Fei, Y.; Liu, Y.; Li, J.Q.; Chen, C. Automated Pixel-Level Pavement Crack Detection on 3D Asphalt Surfaces Using a Deep-Learning Network. *Comput. Civ. Infrastruct. Eng.* **2017**, *32*, 805–819. [CrossRef]
26. Nhat-Duc, H.; Nguyen, Q.-L.; Tran, V.-D. Automatic recognition of asphalt pavement cracks using metaheuristic optimized edge detection algorithms and convolution neural network. *Autom. Constr.* **2018**, *94*, 203–213. [CrossRef]
27. Bi, C.; Fu, B.; Chen, J.; Zhao, Y.; Yang, L.; Duan, Y.; Shi, Y. Machine learning based fast multi-layer liquefaction disaster assessment. *World Wide Web* **2018**, *22*, 1935–1950. [CrossRef]
28. Li, H.; Xu, Q.; He, Y.; Fan, X.; Li, S. Modeling and predicting reservoir landslide displacement with deep belief network and EWMA control charts: A case study in Three Gorges Reservoir. *Landslides* **2019**, *17*, 693–707. [CrossRef]
29. Li, H.; Xu, Q.; He, Y.; Deng, J. Prediction of landslide displacement with an en-semble-based extreme learning machine and copula models. *Landslides* **2018**, *15*, 2047–2059. [CrossRef]
30. Huang, L.; Li, J.; Hao, H.; Li, X. Micro-seismic event detection and location in underground mines by using Convolutional Neural Networks (CNN) and deep learning. *Tunn. Undergr. Space Technol.* **2018**, *81*, 265–276. [CrossRef]
31. Nuttall, J. Estimating spatial correlations from CPT data using neural networks and random fields. In *Numerical Methods in Geotechnical Engineering: Proceedings of the 9th European Conference on Numerical Methods in Geotechnical Engineering, Porto, Portugal, 25–27 June 2018*; Informa UK Limited: Porto, Portugal, 2018; pp. 713–717.
32. Huang, F.; Zhang, J.; Zhou, C.; Wang, Y.; Huang, J.; Zhu, L. A deep learning al-gorithm using a fully connected sparse autoencoder neural network for landslide susceptibility prediction. *Landslides* **2020**, *17*, 217–229. [CrossRef]
33. Hua, Y.; Wang, X.; Li, Y.; Xu, P.; Xia, W. Dynamic Development of Landslide Susceptibility Based on Slope Unit and Deep Neural Networks. *Landslides* **2020**, *18*, 1–22. [CrossRef]
34. Azmoon, B.; Biniyaz, A.; Liu, Z. Image-Data Driven Slope Stability Analysis for Preventing Landslides Using Deep Learning. *IEEE Access* **2021**. manuscript submitted for publication.
35. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
36. Gu, J.; Wang, Z.; Kuen, J.; Ma, L.; Shahroudy, A.; Shuai, B.; Liu, T.; Wang, X.; Wang, G.; Cai, J.; et al. Recent advances in convolutional neural networks. *Pattern Recognit.* **2018**, *77*, 354–377. [CrossRef]
37. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv* **2015**, arXiv:1502.03167.
38. Lui, H.W.; Chow, K.L. Multiclass classification of myocardial infarction with convolutional and recurrent neural networks for portable ECG devices. *Inform. Med. Unlocked* **2018**, *13*, 26–33. [CrossRef]
39. Guo, Y.; Liu, Y.; Oerlemans, A.; Lao, S.; Wu, S.; Lew, M.S. Deep learning for visual understanding: A review. *Neurocomputing* **2016**, *187*, 27–48. [CrossRef]
40. Jia, Y.; Shelhamer, E.; Donahue, J.; Karayev, S.; Long, J.; Girshick, R.; Guadarrama, S.; Darrell, T. Caffe: Convolutional architecture for fast feature embedding. In Proceedings of the 22nd ACM International Conference on Multimedia, Orlando, FL, USA, 3–7 November 2014; pp. 675–678.
41. Cengil, E.; Cinar, A.; Ozbay, E. Image classification with caffe deep learning framework. In Proceedings of the 2017 International Conference on Computer Science and Engineering (UBMK), Antalya, Turkey, 5–8 October 2017; pp. 440–444.
42. Rawat, W.; Wang, Z. Deep convolutional neural networks for image classifica-tion: A comprehensive review. *Neural Comput.* **2017**, *29*, 2352–2449. [CrossRef] [PubMed]
43. Özsert Yiğit, G.; Özyildirim, B.M. Comparison of convolutional neural network models for food image classification. *J. Inf. Telecommun.* **2018**, *2*, 347–357. [CrossRef]
44. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
45. Lathuiliere, S.; Mesejo, P.; Alameda-Pineda, X.; Horaud, R. A Comprehensive Analysis of Deep Regression. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *42*, 2065–2081. [CrossRef] [PubMed]
46. BAIR. Euclidean Loss Layer. Available online: http://caffe.berkeleyvision.org/doxygen/classcaffe_1_1EuclideanLossLayer.html#details (accessed on 28 June 2021).
47. Peck, R.B.; Hanson, W.E.; Thornburn, T.H. Foundation Engineering. *Soil Sci.* **1953**, *75*, 329. [CrossRef]
48. Swiss Standard, S.N. In 670 010b. In *Characteristic Coefficients of Soils*; Association of Swiss Road and Traffic Engineers: Zurich, Switzerland, 1999.
49. Gonzalez, R.C.; Woods, R.E.; Masters, B.R. *Digital Image Processing*, 3rd ed.; Pearson International Edition: London, UK, 2008.
50. Wang, Y.; Chen, Q.; Zhang, B. Image enhancement based on equal area dualistic sub-image histogram equalization method. *IEEE Trans. Consum. Electron.* **1999**, *45*, 68–75. [CrossRef]
51. Albataineh, N. Slope Stability Analysis Using 2D and 3D Methods. Ph.D. Thesis, University of Akron, Akron, OH, USA, 2006.
52. Duncan, J.; Wright, S. The accuracy of equilibrium methods of slope stability analysis. *Eng. Geol.* **1980**, *16*, 5–17. [CrossRef]
53. Cala, M.; Flisiak, J. *Slope Stability Analysis with FLAC and Limit Equilibrium Methods*; CRC Press: Boca Ranton, FL, USA, 2020; pp. 111–114.
54. Cheng, Y.; Länsivaara, T.; Wei, W. Two-dimensional slope stability analysis by limit equilibrium and strength reduction methods. *Comput. Geotech.* **2007**, *34*, 137–150. [CrossRef]
55. Bishop, A.W. The use of the Slip Circle in the Stability Analysis of Slopes. *Géotechnique* **1955**, *5*, 7–17. [CrossRef]
56. Fredlund, D.G.; Krahn, J. Comparison of slope stability methods of analysis. *Can. Geotech. J.* **1977**, *14*, 429–439. [CrossRef]

57. Abramson, L.W.; Lee, T.S.; Sharma, S.; Boyce, G.M. *Slope Stability and Stabilization Methods*; John Wiley & Sons: Hoboken, NJ, USA, 2001.
58. Wright, S.G.; Kulhawy, F.H.; Duncan, J.M. Accuracy of Equilibrium Slope Stability Analysis. *J. Soil Mech. Found. Div.* **1973**, *99*, 783–791. [CrossRef]
59. Aryal, K.P. Slope Stability Evaluations by Limit Equilibrium and Finite Element Methods. Ph.D. Thesis, Norwegian University of Science and Technology, Trondheim, Norway, 2006.
60. Whitman, R.V.; Bailey, W.A. Use of Computers for Slope Stability Analysis. *J. Soil Mech. Found. Div.* **1967**, *93*, 475–498. [CrossRef]
61. Janbu, N. Slope stability computations. *Int. J. Rock Mech. Min. Sci. Géoméch. Abstr.* **1975**, *12*, 67. [CrossRef]
62. Janbu, N. Earth pressure and bearing capacity calculations by generalized procedure of slices. In Proceedings of the 4th International Conference SMFE, London, UK, 12–24 August 1957; pp. 207–212.
63. Janbu, N. *Slope Stability Computations*; Soil Mechanics and Foundation Engineering Report; Technical University of Norway: Trondheim, Norway, 1968.
64. Janbu, N. Application of composite slip surface for stability analysis. *Proc. Eur. Conf. Stab. Earth Slopes* **1954**, *3*, 43–49.
65. Olivas, E.S.; Guerrero, J.D.M.; Martinez-Sober, M.; Magdalena-Benedito, J.R.; Serrano, L. *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques*; IGI Global: Hershey, PA, USA, 2019.
66. Yosinski, J.; Clune, J.; Bengio, Y.; Lipson, H. How Transferable Are Features in Deep Neural Networks? *arXiv* **2014**, arXiv:1411.1792.
67. Zeiler, M.D. Adadelta: An adaptive learning rate method. *arXiv* **2012**, arXiv:1212.5701.