Grade Crossing Monitoring Using Deep Learning

Gasser Ali, Ph.D.
Principal Investigator
Civil Engineering Department
University of Texas Rio Grande Valley

Constantine Tarawneh, Ph.D.
Director, UTCRS, Co-Principal Investigator
Mechanical Engineering Department
University of Texas Rio Grande Valley

Felix Chavez
Undergraduate Research Assistant
Computer Science Department
University of Texas Rio Grande Valley

Darren Espinoza
Undergraduate Research Assistant
Mechanical Engineering Department
University of Texas Rio Grande Valley

A Report on Research Sponsored by

University Transportation Center for Railway Safety (UTCRS)

University of Texas Rio Grande Valley (UTRGV)

September 2024

Technical Report Documentation Page

1. Report No. UTCRS-UTRGV-O6CY23	2. Government Accession No.	3. Recipient's Catalog No.

4. Title and Subtitle Grade Crossing Monitoring Using Deep Learning	5. Report Date September 25, 2024
	6. Performing Organization Code UTCRS-UTRGV
7. Author(s) Gasser Ali, Constantine Tarawneh, Darren Espinoza, and Felix Chavez	8. Performing Organization Report No. UTCRS-UTRGV-O6CY23
9. Performing Organization Name and Address University Transportation Center for Railway Safety (UTCRS) University of Texas Rio Grande Valley (UTRGV) 1201 W. University Dr. Edinburg, TX 78539	10. Work Unit No. (TRAIS)
	11. Contract or Grant No. 69A3552348340
12. Sponsoring Agency Name and Address U.S. Department of Transportation (USDOT) University Transportation Centers Program 1200 New Jersey Ave. SE Washington, DC, 20590	13. Type of Report and Period Covered Project Report June 1, 2023 – August 31, 2024
	14. Sponsoring Agency Code USDOT UTC Program

15. Supplementary Notes

16. Abstract

Railway crossings are critical elements of railway safety due to the heightened risk of collisions. Transportation agencies and researchers are continuously working to enhance safety at railway crossings with better operating procedures and equipment to avoid accidents. Many innovative methods have been proposed to detect hazards at crossings and rail tracks using technologies such as sensors, computer vision, depth cameras, and many others. However, there is still a need to develop a holistic approach that is robust and generalizable to the many conditions and hazards related to grade crossing accidents. This project investigates Artificial Intelligence (AI) and Deep Learning (DL) models to monitor grade crossings and detect various hazardous conditions such as vehicles, pedestrians, cyclists, animals, warning lights, and others. To achieve that, the methodology consists of (1) collecting visual data of railway crossings; (2) labeling the data for training; and (3) developing a computer vision model using deep learning that can detect hazardous conditions at railway crossings. Ultimately, the outcomes of this research support modernizing and improving safety at crossings.

17. Key Words Computer Vision, Automatic Incident Detection, Artificial Intelligence, Neural Networks	18. Distribution Statement This report is available for download from https://www.utrgv.edu/railwaysafety/research/operations/index.htm

19. Security Classification (of this report) None	20. Security Classification (of this page) None	21. No. of Pages 19	22. Price

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AI	Artificial Intelligence
CNN	Convolutional Neural Network
CV	Computer Vision
DL	Deep Learning
FN	False Negative
FP	False Positive
NHTSA	National Highway Traffic Safety Administration
TN	True Negative
TP	True Positive
USDOT	U.S. Department of Transportation
UTC	University Transportation Center

Disclaimer

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated under the sponsorship of the U.S. Department of Transportation's University Transportation Centers Program, in the interest of information exchange. The U.S. Government assumes no liability for the contents or use thereof.

Acknowledgements

The authors want to acknowledge the University Transportation Center for Railway Safety (UTCRS) at UTRGV for the financial support provided to perform this study through the USDOT UTC Program under Grant No. 69A3552348340. The authors also wish to acknowledge partial financial support provided by the U.S. National Science Foundation (NSF) under Award No. 2112650.

Introduction

Safety at rail crossings is critical to avoid accidents between trains, vehicles, and pedestrians. The National Highway Traffic Safety Administration (NHTSA) recorded 1,600 vehicle-train and 500 human-train accidents at railroad crossings in 2020 [1]. There is an urgent need to reduce these high accident rates and bolster crossing safety. Transport authorities are continuously implementing and investigating effective policies and technologies to enhance crossing safety. Toward this goal, several innovative technologies have been suggested by researchers to support rail safety. These include CCTV systems utilizing computer vision to monitor vehicle and pedestrian movements at crossings [2–4], along with depth or stereo cameras [5,6]. Nevertheless, there remains a demand for a versatile system capable of adapting to different crossings, identifying various hazards, and operating under varied weather and light conditions. Artificial Intelligence (AI) and Computer Vision (CV), particularly through Deep Learning (DL), appear to be promising techniques to meet these challenges. DL has catalyzed significant scientific advancements in recent years across various fields including visual object recognition, natural language processing, and more [7]. Convolutional Neural Networks (CNNs), a category of deep learning models, have proven effective in analyzing images, videos, and sounds. In rail safety, CNNs have been applied to detect falls, slips, and trips at stations [8], automate stopping for autonomous trains [9], control level crossings by detecting approaching trains [10], and monitor traffic at railway crossings [11]. However, challenges persist in deploying deep learning for railway safety due to data scarcity, limited data sources, the necessity for real-time processing, task generalization, and system validation [12].

Goal

This research explores the use of AI, specifically DL through CNN architectures, for monitoring grade crossings and identifying potential hazards like vehicles, pedestrians, animals, warning signals, and the status of safety arms. There is a pressing need for a scalable AI model that functions across various grade crossings under different conditions linked to accidents and near-misses. This objective is realized by developing a CNN model trained on a dataset of rail crossing images, which were collected and annotated by the researchers. The results aim to bolster the role of AI in enhancing railway safety. The model described herein offers a reliable and economical approach that can leverage standard cameras at grade crossings to continuously monitor safety risks.

Methodology

The goal of this research is achieved through a structured multi-step methodology as illustrated in Figure 1. The steps are as follows: (1) Data Collection: Visual data from railway crossings are sourced from live stream feeds across diverse locations, subjected to varying environmental conditions and times. This raw data is processed using a convolutional neural network (CNN) to remove superfluous images, setting the stage for model training. (2) Data Labeling: The images are manually tagged to identify obstacles and conditions, utilizing data labeling software to enhance the efficiency of manual labeling and manage the database effectively. (3) Model Training: A computer vision model is constructed with TensorFlow and Keras in Python, trained on 80% of the annotated data. (4) Model Validation: The model undergoes validation with the remaining 20% of the labeled data. Further details are provided in the subsequent subsections.

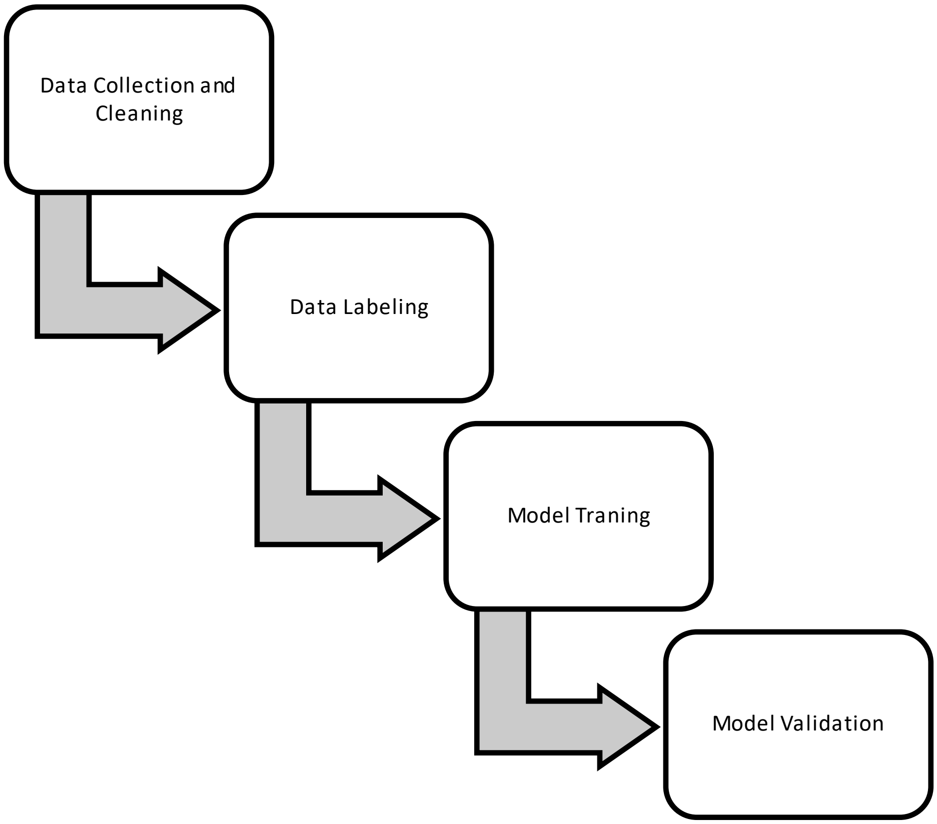


Figure 1: Methodology

Data Collection

Training DL models necessitates an extensive array of images, divided into training and testing/validation sets. The effectiveness of a model is significantly influenced by both the volume and integrity of the data used in its training. Manually gathering a vast dataset can be labor-intensive. To address this, the authors employed an automated tool to download live video streams of grade crossings across the US, using the Python package "yt-dlp" [13]. Live feeds broadcasted on YouTube by "Virtual RailFan, Inc." were archived [14]. These streams were saved as video files with a resolution of 640×360 pixels. A script was then used to select and convert every fifth frame from the videos into individual images. This method generates datasets encompassing thousands of images. Two representative images from this collection are displayed in Figure 2.

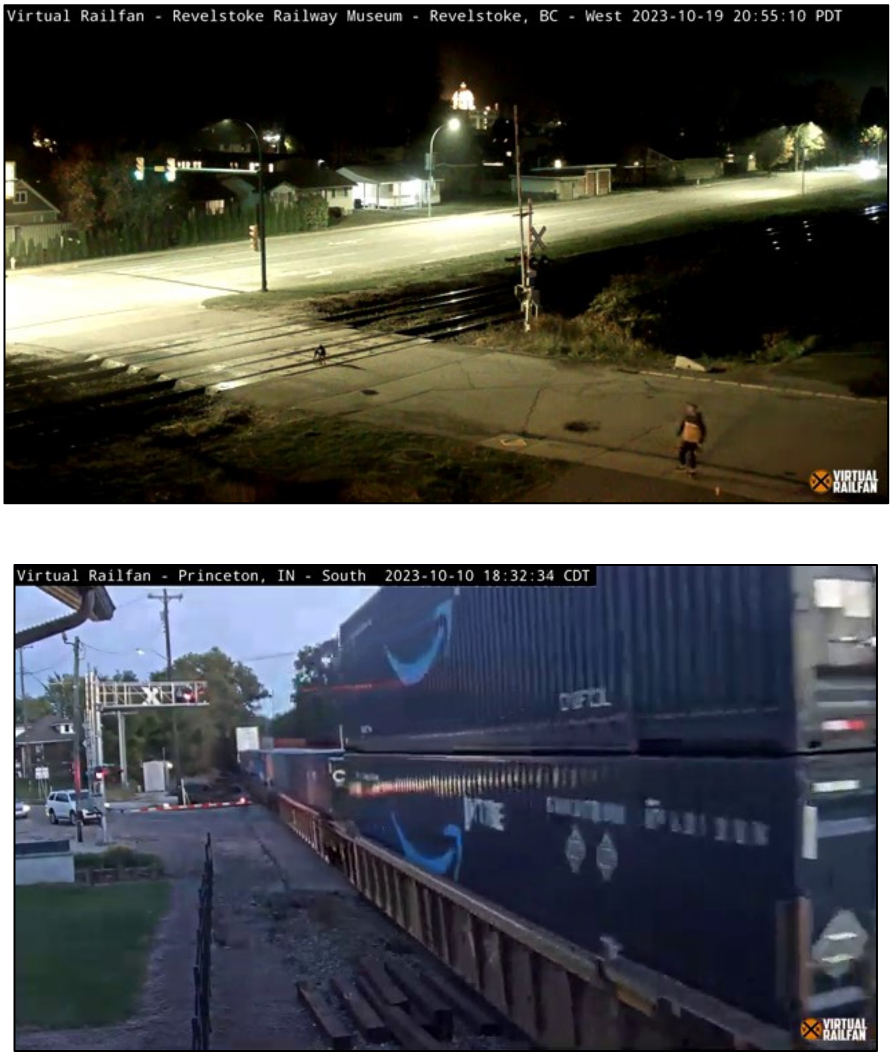

Virtual Railfan - Revelstoke Railway Museum - Revelstoke, BC - West 2023-10-19 20:55:10 PDT
VIRTUAL RAILFAN
Virtual Railfan - Princeton, IN - South 2023-10-10 18:32:34 CDT
VIRTUAL RAILFAN

Figure 2: Examples of collected images

Data Cleaning

Given that many of the collected images are redundant, showing empty grade crossings without any significant activity, it is crucial to methodically refine the dataset. This is accomplished using a CNN model called "Image Duplicator (Imagededup)", which identifies and eliminates duplicate photos with a similarity threshold of 95%. This cleaning process produces a refined dataset of unique images that represent various grade crossings under different conditions, capturing a range of potential hazards.

Data Labeling

Data labeling consists of manually recording relevant items and/or activities in images in preparation for model training and validations. The labels marked in this process represent the target data for the model. To develop the model, the data is handled as a binary multilabel classification problem. The labels considered are shown in Table 1. The labels were chosen by the authors to indicate the condition of the rail crossings as related to trains, vehicles, pedestrians, animals, and warning systems. Each image may contain one or more labels or none. The CNN has binary outputs associated with each of the labels in Table 1. The data labeling process was manually performed for each image. A data labeling software, "Label Studio", is used to facilitate label input and database management [15]. The output of this process is a database of the filenames and associated labels for each image in the collected dataset. After this step, the dataset contained 1,364 labeled images.

Model Architecture

The model is developed as a deep CNN using TensorFlow and Keras in Python [16,17]. The CNN consists of a sequence of layers as shown in Figure 3. The convolutional layers apply a convolution operation on two-dimensional inputs, which involves sliding a window over the input data. This operation enables the model to detect patterns in images. Pooling layers are introduced between convolution layers. Pooling layers reduce the dimensionality of the outputs of the convolution layers by taking the maximum values from adjacent pixels. One "Flatten Layer" is inserted after the last pooling layer to "flatten" its two-dimensional input into one-dimensional for the following "dense" layer of neurons. Dropout layers help improve the generalization of the model by randomly setting a fraction of input units to zero during training, which addresses the problem of overfitting. Overall, the developed model has 10,263,370 trainable parameters. The inputs of the model are 400 × 400 images, and the model has ten outputs representing each label. Model training uses the Adam optimizer [18] with a binary cross-entropy loss function.

Table 1: Label descriptions

Label #	Name
1	Animals
2	Grade Crossing Gate Down
3	Train
4	Red Light on Grade Crossing
5	People on Grade Crossing
6	Vehicle on Grade Crossing
7	Grade Crossing
8	Train on Grade Crossing
9	Vehicle Waiting for Train
10	Animal on Grade Crossing

Data Splitting for Training and Testing

The data from the labeling set includes 1,364 labeled images. The data is subjected to an 80/20 testing/validation split after random shuffling. Accordingly, 1,092 images are used for training, and 272 images are used for validation. The purpose of the testing/validation split is to measure the performance of the model on data that has not been used for model training. In other words, the model is tested on images that it has not seen and learned from before. The validation step evaluates how the model would perform after deployment.

Data Pre-Processing and Augmentation

During model training, the images are automatically fed into a pipeline that performs resizing and data augmentation. Images are all resized to 400 × 400 pixels. The data augmentation process artificially creates new data for the model by leveraging old data. Data augmentation performs random transformations to the images, which include rotation, size changes, shear, zoom, and horizontal flips. This process improves the generalization of the model during training.

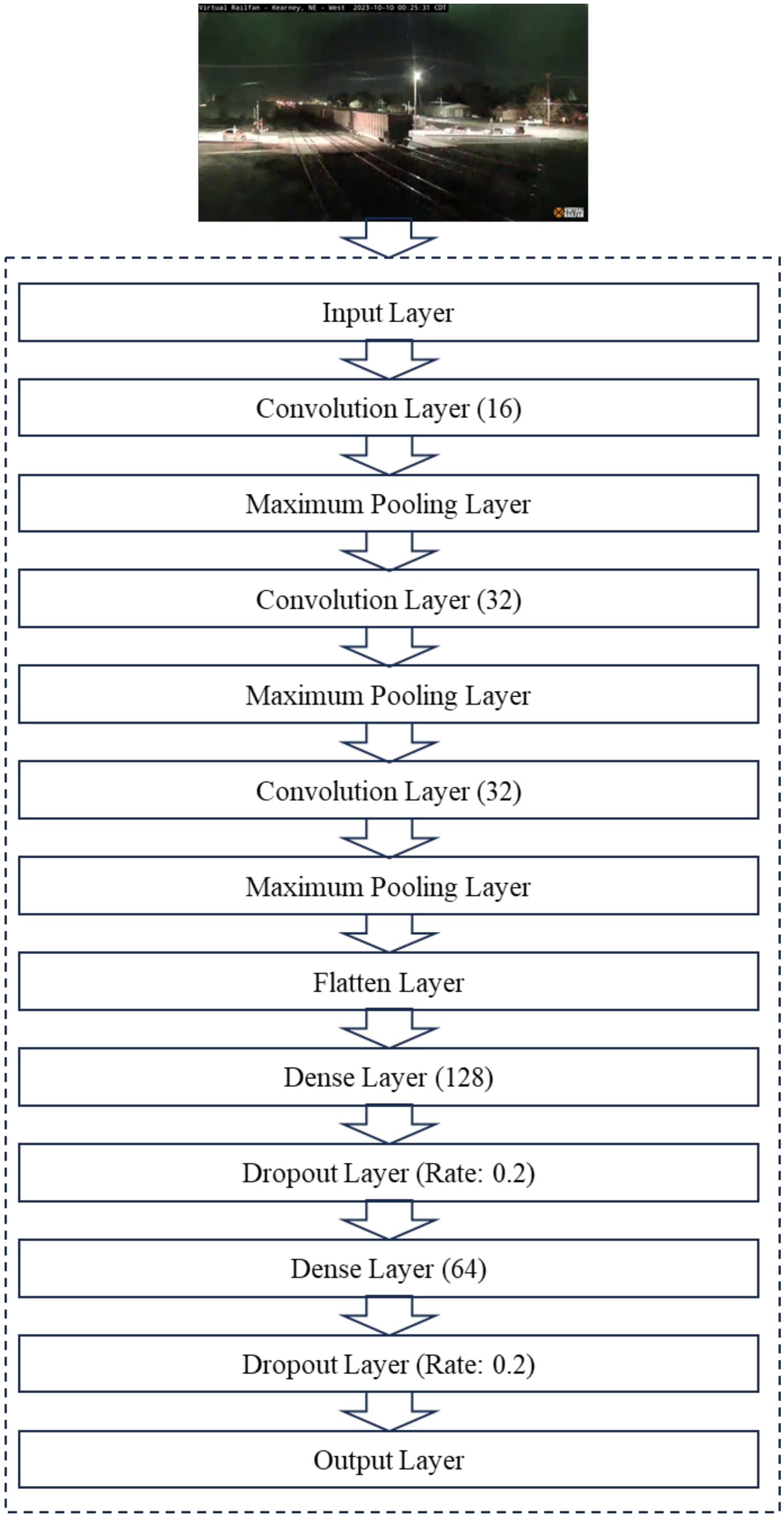


Figure 3: Model architecture

Results and Analysis

The following paragraphs present the analysis of the data collection to explore the quality of the dataset, followed by the results of the model training and validation.

Analysis of Collected Data

The labeled data used for training and validation includes 1,364 unique images. The number of images by label is shown in Figure 4. It should be noted that the sum of the numbers in the figure does not represent the total number of images because each image may be tagged by more than one label. A limitation of the dataset is that it does not include many images of animals or pedestrians due to their low number compared to other items and events. Other labels are well represented in the dataset.

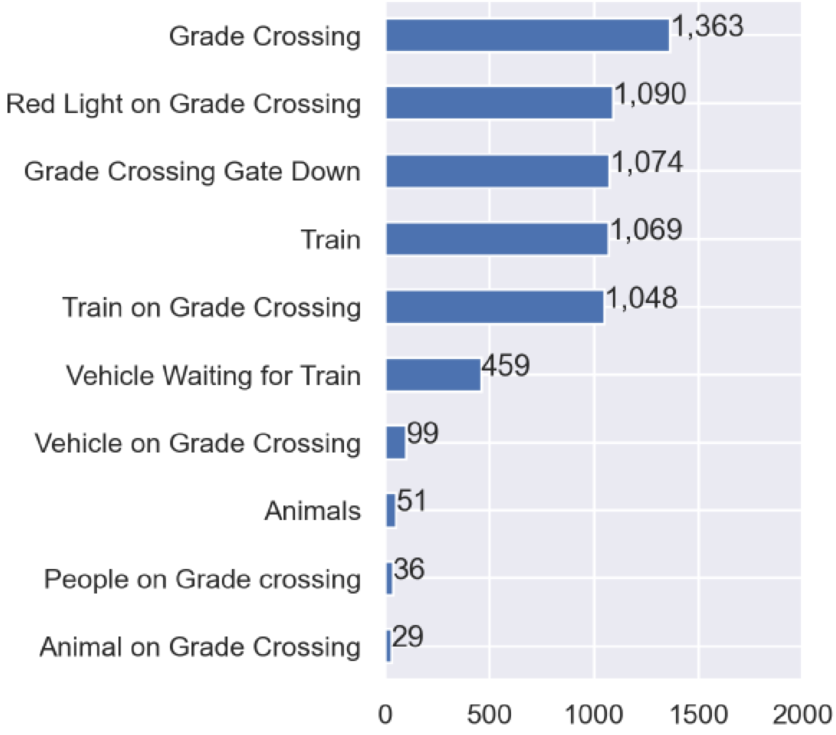
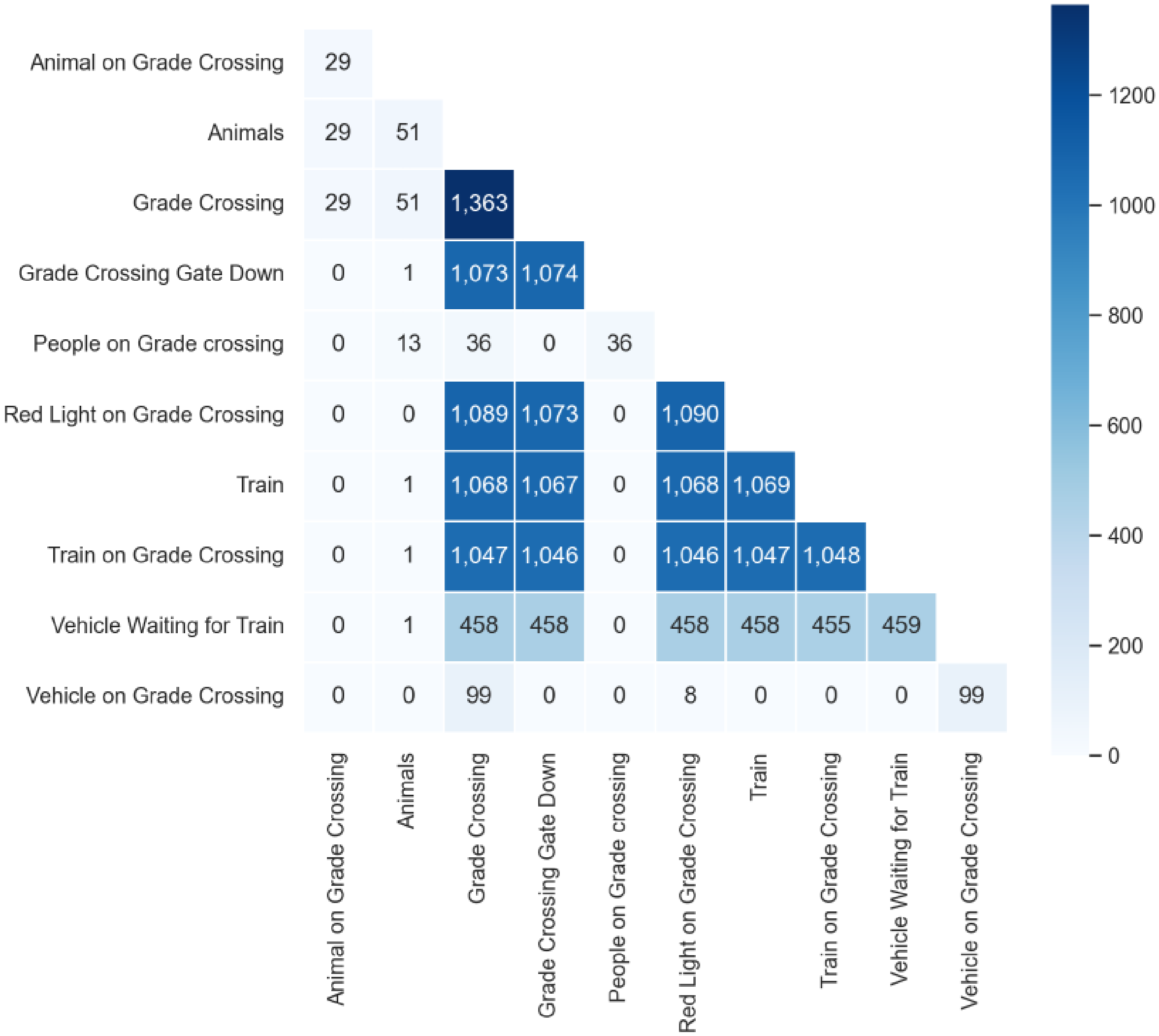


Figure 4: Number of images by label

The number of images associated with each combination of labels is shown in Figure 5. It can be noted that the numbers on the diagonal represent the same values shown in Figure 4. Otherwise, the numbers represent the number of images where two labels are positive at the same time. For example, there are 1,073 images where the red lights at the grade crossing are active and the grade crossing gate is closed at the same time. This number is high because the two events are intuitively correlated. However, the figure also shows there are deficiencies in the number of photos associated with pedestrians and animals at crossings with other labels.



	Animal on Grade Crossing	Animals	Grade Crossing	Grade Crossing Gate Down	People on Grade crossing	Red Light on Grade Crossing	Train	Train on Grade Crossing	Vehicle Waiting for Train	Vehicle on Grade Crossing
Animal on Grade Crossing	29									
Animals	29	51								
Grade Crossing	29	51	1,363							
Grade Crossing Gate Down	0	1	1,073	1,074						
People on Grade crossing	0	13	36	0	36					
Red Light on Grade Crossing	0	0	1,089	1,073	0	1,090				
Train	0	1	1,068	1,067	0	1,068	1,069			
Train on Grade Crossing	0	1	1,047	1,046	0	1,046	1,047	1,048		
Vehicle Waiting for Train	0	1	458	458	0	458	458	455	459	
Vehicle on Grade Crossing	0	0	99	0	0	8	0	0	0	99

Figure 5: Adjacency matrix of label combinations

To further analyze the dataset, the correlation between the labels is shown in Figure 6. A value of +1 indicates the two events are perfectly correlated, while a value of −1 indicates a perfect negative relationship and a value of zero indicates no relationship. For instance, there is a high correlation between the red lights, the grade crossing being closed, and a train showing in the image.

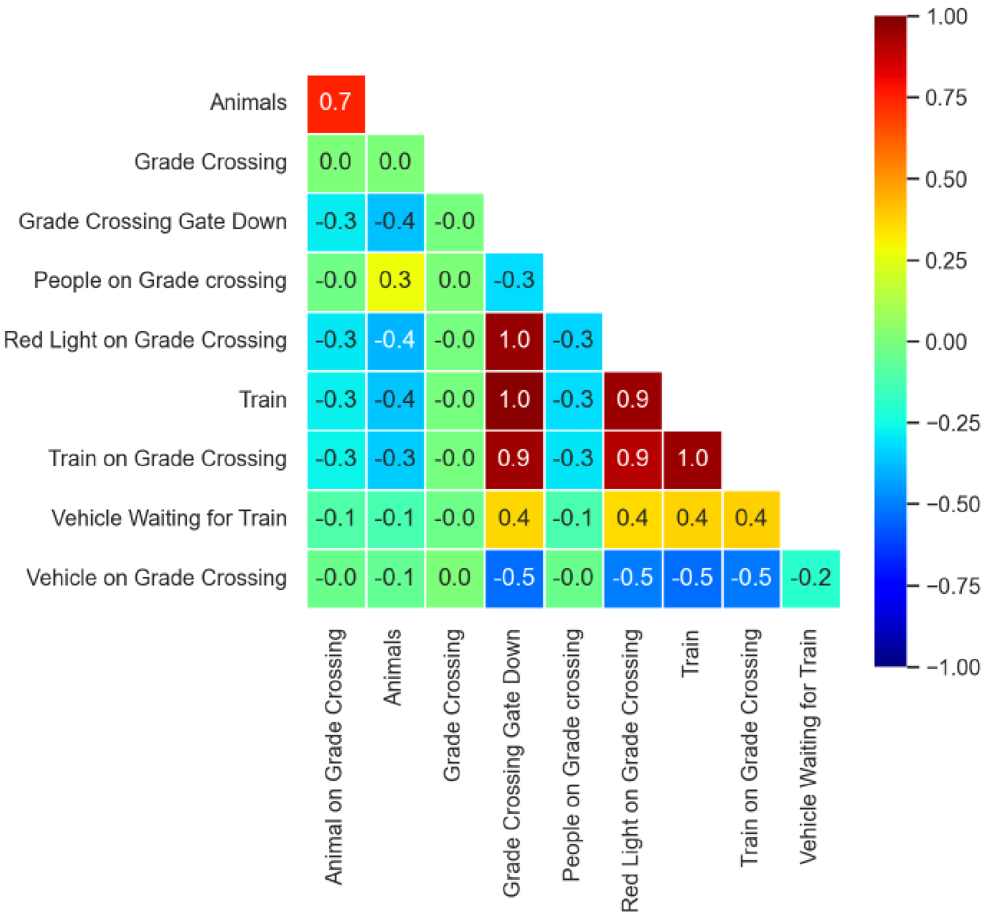


Figure 6: Correlation between labels

Overall, this analysis shows that the quality of the dataset is acceptable. However, it can be greatly improved by adding more photos of pedestrians and animals, among other items such as cyclists for example. This need will be addressed in future research.

Model Training Results

The model was trained for 3000 epochs. As previously noted, the model was trained after an 80/20% training/validation data split. The history of the training is shown in Figure 7 as related to training and validation losses using binary cross-entropy, and in Figure 8 as related to accuracy. The figures show that the number of epochs of 3000 is a suitable stopping criterion. Furthermore, there is minimal overtraining for the model.

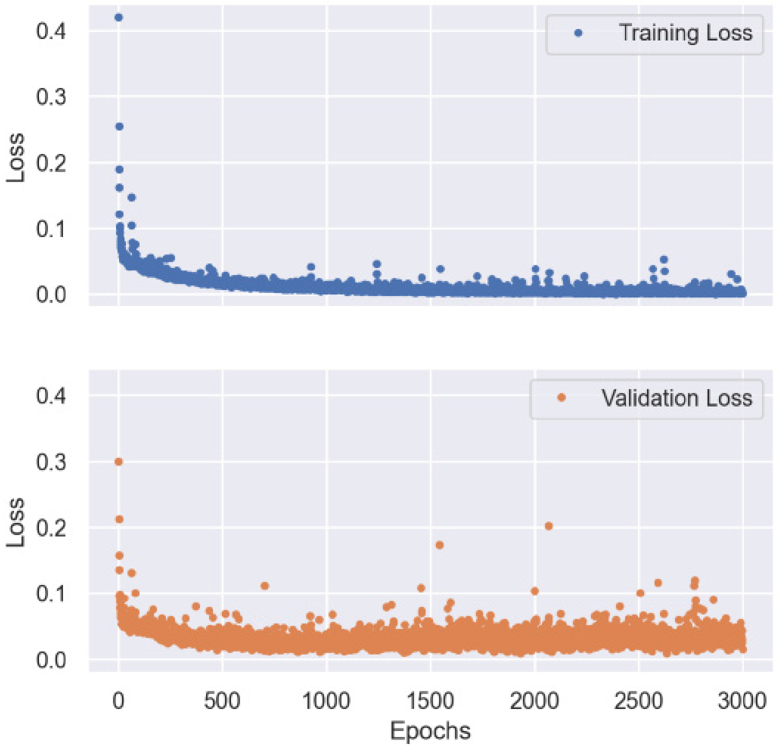


Figure 7: Training and validation losses

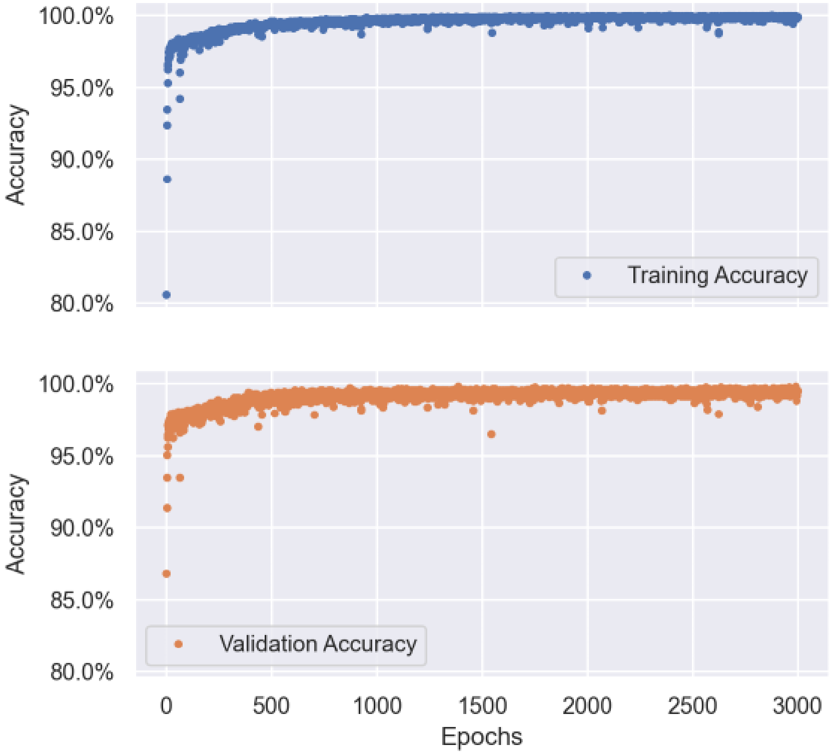


Figure 8: Training and validation accuracies

Model Validation Results

As previously noted, 272 images are used for model validation. This set of images is not used for training the model. As such, validation evaluates how the model would perform when deployed. The metrics related to the performance of the model for each image in the validation set are shown in Table 2. The metrics shown in the table are True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). The metrics are further analyzed in Table 2 which shows the accuracy, precision, and recall. All the metrics show that the model achieved a high accuracy in classifying all types of events and hazards. For the goal of improving safety at grade crossings, the most important metrics to note should be the FNs and the recall, because it is critical that the model does not incorrectly indicate there is no train or vehicles on crossings. The recall of the model for detecting vehicles on crossings is 95%. However, it must be noted that the performance results of the model are preliminary and must be further enhanced through more data.

Table 2: Validation results

Label	***TP***	***TN***	***FP***	***FN***	**Accuracy%** $\left(\frac{TP+TN}{P+N}\right)$	**Precision%** $\left(\frac{TP}{TP+FP}\right)$	**Recall%** $\left(\frac{TP}{TP+FN}\right)$
Animal on Grade Crossing	11	261	0	0	100.00%	100.00%	100.00%
Animals	12	260	0	0	100.00%	100.00%	100.00%
Grade Crossing	272	0	0	0	100.00%	100.00%	100.00%
Grade Crossing Gate Down	203	67	0	2	99.26%	100.00%	99.02%
People on Grade crossing	6	266	0	0	100.00%	100.00%	100.00%
Red Light on Grade Crossing	206	64	0	2	99.26%	100.00%	99.04%
Train	201	68	1	2	98.90%	99.50%	99.01%
Train on Grade Crossing	195	75	2	0	99.26%	98.98%	100.00%
Vehicle Waiting for Train	83	189	0	0	100.00%	100.00%	100.00%
Vehicle on Grade Crossing	19	250	2	1	98.90%	90.48%	95.00%

Conclusion

The goal of this research project was to investigate the application of AI for grade crossing safety. This goal was achieved by creating a CNN model and training it using a dataset collected and labeled by the authors. The performance of the model is satisfactory, with a precision reaching 95% for detecting vehicles on grade crossing tracks. However, more data is needed to ensure the reliability of the model. As such, the limitation of this research is the lack of sufficient training images for pedestrians, animals, and cyclists, to ensure better training and to validate the performance of the model. This limitation is being addressed in ongoing research.

Ultimately, the outcomes of this research aim to improve safety at grade crossings by safeguarding lives, avoiding costly accidents, and reducing network downtime. Specifically, the developed model can be adapted to embedded systems with cameras or CCTV to monitor grade crossings and communicate with Positive Train Control (PTC) systems. In addition, long-term monitoring results of grade crossing can help identify high-risk crossings and suggest improvements to crossing safety policies.

References

[1] NHTSA, 2023, "Railroad Crossing | NHTSA" [Online]. Available: https://web.archive.org/web/20230331141845/https://www.nhtsa.gov/campaign/railroad-crossing. [Accessed: 07-May-2023].

[2] Sheikh, Y. A., Zhai, Y., Shafique, K., and Shah, M. A., 2004, "Visual Monitoring of Railroad Grade Crossing," E.M. Carapezza, ed., Orlando, FL, p. 654.

[3] Shin, D., Jin, J., and Kim, J., 2021, "Enhancing Railway Maintenance Safety Using Open-Source Computer Vision," Journal of Advanced Transportation, 2021, p. e5575557.

[4] Zhang, Z., Trivedi, C., and Liu, X., 2018, "Automated Detection of Grade-Crossing-Trespassing near Misses Based on Computer Vision Analysis of Surveillance Video Data," Safety Science, 110, pp. 276–285.

[5] Hosotani, D., Yoda, I., and Sakaue, K., 2009, "Development and Long-Term Verification of Stereo Vision Sensor System for Controlling Safety at Railroad Crossing," *Computer Vision Systems*, M. Fritz, B. Schiele, and J.H. Piater, eds., Springer, Berlin, Heidelberg, pp. 154–163.

[6] Yoda, I., Sakaue, K., and Hosotani, D., 2006, "Multi-Point Stereo Camera System for Controlling Safety at Railroad Crossings," *Fourth IEEE International Conference on Computer Vision Systems (ICVS'06)*, pp. 51–51.

[7] LeCun, Y., Bengio, Y., and Hinton, G., 2015, "Deep Learning," Nature, 521(7553), pp. 436–444.

[8] Alawad, H., Kaewunruen, S., and An, M., 2020, "A Deep Learning Approach Towards Railway Safety Risk Assessment," IEEE Access, 8, pp. 102811–102832.

[9] Etxeberria-Garcia, M., Labayen, M., Zamalloa, M., and Arana-Arexolaleiba, N., 2020, "Application of Computer Vision and Deep Learning in the Railway Domain for Autonomous Train Stop Operation," *2020 IEEE/SICE International Symposium on System Integration (SII)*, pp. 943–948.

[10] Murshed, R. U., Dhruba, S. K., Bhuian, Md. T. I., and Akter, Mst. R., 2022, "Automated Level Crossing System: A Computer Vision Based Approach with Raspberry Pi Microcontroller," *2022 12th International Conference on Electrical and Computer Engineering (ICECE)*, pp. 180–183.

[11] Guo, F., Wang, Y., and Qian, Y., 2022, "Computer Vision-Based Approach for Smart Traffic Condition Assessment at the Railroad Grade Crossing," Advanced Engineering Informatics, 51, p. 101456.

[12] Oh, K., Yoo, M., Jin, N., Ko, J., Seo, J., Joo, H., and Ko, M., 2022, "A Review of Deep Learning Applications for Railway Safety," Applied Sciences, 12(20), p. 10572.

[13] yt-dlp, 2023, "Yt-Dlp/Yt-Dlp: A Youtube-Dl Fork with Additional Features and Fixes."

[14] Virtual Railfan, Inc., 2023, "Virtual Railfan, Inc. – We Bring the Trains to You!" [Online]. Available: https://virtualrailfan.com/. [Accessed: 03-Dec-2023].

[15] Label Studio, 2023, "Open Source Data Labeling," Label Studio [Online]. Available: https://labelstud.io/. [Accessed: 03-Dec-2023].

[16] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mane, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viegas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X., 2016, "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems."

[17] Gulli, A., and Pal, S., 2017, *Deep Learning with Keras*, Packt Publishing Ltd.

[18] Kingma, D. P., and Ba, J., 2017, "Adam: A Method for Stochastic Optimization."