Project SAFE: Safe path Approximation For Elementary schools

Harsha Raja Shivakumar
Jong Sung Park
Maithreyi Manur Narasimha Prabhu
Vansh Shah

Indiana University, Bloomington
CSCI-B657 Computer Vision
Final report
Inspired by Prof. Katie Siek from IUB
http://github.iu.edu/jp109/CV_final_project

Abstract

Project SAFE aims to calculate the safety score and a safe path to an elementary school by object detection / classification of street view images. The coordinates within the 1-mile radius of an elementary school are extracted and street view images are downloaded accordingly. The images are fed to the models to calculate the safety score per coordinate, which is used to calculate a safe path. Results show that the program provides a better path a student can take to walk to school and compares the safety score between the different classes of elementary schools.

1. Introduction

According to a 2015 Data from NHTSA, 'on average 3 children were killed and an estimated 487 children were injured every day in the United States in traffic crashes'.[1] Children can be more vulnerable to cars due to their body size and unexpected maneuver. We can all agree that children should have a safe path to use at least on a walk to the school day.

The Safe Routes to School listserve argues that 1 mile is the distance older elementary students can walk to school.[2] As depicted in Figure 1, there can be an unsafe path for children to walk, even though the path is suggested by google as a walking path. A safe path can exist from a house to school, but it wouldn't make sense if you had to walk all the way around the block looking for a road with sidewalks, stop signs, crosswalks, etc.

There have been projects related to this kind of safety, like Project Sidewalk[3], which determines how safe the region is using crowdsourced labels of street view images. WalkIT Arizona[4] is another project that aims to improve the walking experience. Though each project has been improving the outdoor activity of the physically disabled and general walking, from our understanding, there has been no project that specifically aims at improving students walking path to school.

Our model aims to solve this problem by getting the street view image of every coordinate and calculating the safety score. This safety score can be used as a heuristic with the overall distance to find the best path a child could use to walk to school.

A coordinate is divided into 4 images, each image showing 0/90/180/270 angle from street view. Then, we use simple CNN models like Alex Net[5] and ResNet[6] to categorize each image. They are categorized by whether; they have 0/1/2 sides that have sidewalks, have crosswalks/signs and have stop signs or traffic lights or none. The classification results can be used as a factor calculating the safety of a certain coordinate. The traffic of the coordinate is also considered in calculating the score.

Our primary contribution is being the first-time introducing children's walking problem with CNN strategies. Another contribution is that we demonstrate a way to suggest a safer way through using safety scores as a heuristic for finding a walking path. We would also like to indicate that this project was inspired by Professor Katie Siek from the Computer Science Department of Indiana University, Bloomington. The TensorFlow Python code is available at http://github.iu.edu/jp109/CV_final_project.

Figure 1: The right image is a street view in some coordinate within the walking path suggested by google.

2. Related work

AlexNet, The deep convolutional neural network for ImageNet classification is a great approach to start a model. It contains five convolutional layers and three fully-connected layers with ReLU as its activation.[5]. As this model was used for classification in the first place, it serves a great starting point for the project. However, it does have some limitations where the model's accuracy is achieved by careful tweaking of the hyperparameters, and thus has its limitations.

ResNet Deep residual learning provides a way of adding some skip connections and make the layers fit a residual mapping instead of directly fitting the features needed.[6]

The approach adds an additional layer after a set of ResNet blocks(made of a convolutional layer, batch normalization[7] layer, and a ReLU activation layer) to implement this. This model has been proven to greatly reduce errors compared to plain networks. Some of our models show improvement using this method. Details on how the blocks were implemented can be seen in the method section below.

3. Method

Each classification of the image is done by separate models, later combined into a single safety score ranging from 0 to 1. The abstract model structures are depicted in Figure 2. Each street view image was rescaled to have a size of 640 x 300 and values from 0 to 1 instead of 0 to 255.

3.1. Sidewalk detection

The model starts with one convolutional layer followed by max-pooling of kernel size 3. Another convolutional layer is followed including batch normalization[7] layer in between. After one more set of convolutional layers and max pooling, 12 ResNet blocks[6] are added. The input is flattened and passed through two fully connected layers like AlexNet[5] and after a dropout layer, the output layer consists of 3 nodes and softmax as its activation. The rest of the activation functions of the layers should be ReLU.
Each of the 3 nodes of the output layer represents having none/one/two sidewalks at the coordinate location. The results were trained using categorical_crossentropy as its loss function.

Nesterov[8] was chosen as its optimizer since stochastic gradient descent momentum[9] was showing a slower train rate(which would be critical once trained on a large dataset) and Adam[10] had been showing fast but wrong convergence. Additionally, early stopping of train loss was implemented with patience 3.

Data augmentation done on the training data include; brightness of range [0.2, 1.0], vertical flip and zoom range of [0.5, 1.0]. Brightness was used because the street tends to show different colors due to shadows and sunlight, the vertical flip is necessary to consider both sides of the road, and the zoom range was considered due to the variance in the width and height of the sidewalks.

3.2. Crosswalk/sign detection

The model creation was inspired based on the Alexnet[5] architecture and it was modified to suit the input size of 640x300. It consists of 5 convolutional layers and each followed by batch normalization[7] layer and max-pooling of kernel size 3 and strides 2. Then the input is flattened and passed through three fully connected layers. The output layer consists of 2 nodes and softmax as its activation. In the rest of the layers, ReLU is used as the activation function with binary_crossentropy as its loss function. Adam[10] was chosen as the optimizer which showed good convergence.

Data augmentation was done on the training data which includes horizontal flip, the zoom range of [0.5, 0.8], noise, and shear. Zoon range was used to cover the variance of crosswalks. Additional noise was added to the images to cover the noise that could be existing in the images. Shear was added to cover various angles of the crosswalk sign.

3.3. Stop signs/Traffic lights detection

The images were gathered from various sources and then augmented with zoom range of 0.3 and 0.6 and shear range of 0.2. Additionally the dataset was imbalanced and so particular classes were also augmented such that the bias in the dataset got reduced. The loss used here is binary_crossentropy and the activation function is Leaky ReLU. It showed better convergence on Adam optimizer[10].

The models used were constructed based on Alexnet[5]. The models are sequential and after every convolution and pooling layer it is followed by batch normalization[7]. The output layer consists of 2 nodes and softmax as the activation function. Stop signs as well as Traffic light with a decent Precision and Recall are predicted.

3.4. Connected graph creation

From the source coordinate (In this paper we are using Luddy School of Informatics, Indiana University) we get the next 4 coordinates with a 10m distance in 4 directions 0, 90, 180, and 270. The next 4 coordinates are extracted using pygeodesy [11] python library. Here we know the distance, source latitude, and longitude with the direction we get the destination latitude and longitude. Once we get these coordinates we add them to the queue to repeat the same process. We get all the coordinates in a Breadth-First Search (BFS) manner. Each source latitude and longitude become the parent and destination latitude and longitude becomes a child. Then in the next iteration, the destination latitude and longitude become the parent and the process goes on. Each time while getting the coordinates we also get the distance between them using Google Distance Matrix API. We are restricting the coordinates to a 1mile radius from the source coordinate using the haversine formula. The haversine formula is used to calculate the shortest distance between 2 points on the sphere using their latitude and longitude measured along the surface. In this way, we created the connected graph which will be used to find the shortest path between any 2 coordinates.

3.5. Traffic data

Traffic data at any given source and destination coordinates are extracted using Bing routes API [13] in

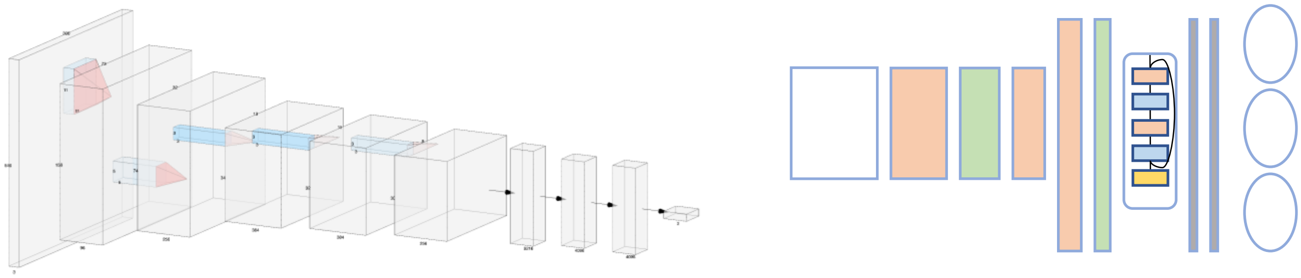

Figure 2 : Each is a simple representation of the model architectures. The left architecture is shared in crosswalk model, cross sign detection model and the street sign/traffic lights model. The right architecture is used in the sidewalk detection model.

driving mode. API gives the categorical values as Unknown, None, Mild, Moderate, and Severe. Given that we are taking coordinates around Bloomington, most of the coordinates had Unknown, None, and Mild values. This will be used as one of the factors while recommending the shortest path.

3.6. Safety score

Using the models provided above, safety scores were calculated from each model. The scores have a range from 0 to 1(where the sidewalk has a range of 0 to 2 because it has 2 sides). Each score was measured so that it best evaluates the current safety condition.

The safety score depending on the sidewalk was calculated to have weight on the probability of having both sidewalks. This is because the dataset showed high false-negative rates on two sidewalk classification, categorizing it as one sidewalk. Details can be seen in the code.

The crosswalk/sign detection models each use the probability of the image having a crosswalk/sign as its safety score. This can be understood as not just considering if the object exists but how visible it is to children. Stop signs and traffic light model follows the same principle.

Each coordinate's safety score was calculated as a squared mean of these scores so that low probabilities will have a higher rate. The overall safety score of a school was calculated again after taking the mean of the coordinate's scores.

3.7. Safe path algorithm

The project simply uses a Search algorithm looking for the safest path, while using some of the values calculated above as heuristics.

Distance between coordinates serves as a part of the heuristic function, as if this is not considered, the algorithm would go around the town just looking for a safe, but not a valid walkable path. Some weight was added so that it would be comparable to other values used in the function.

Safety score was added as a heuristic, but since it evaluates how safe a coordinate is, it was used in a form of 1 – score. This way, it would have a value of 0 is completely safe, and 1 if 100% dangerous, which can be used as a part of heuristic where we use the minimum value in the heap.

Traffic data was also used, with a weight of 0.1. Since Bloomington shows the small difference in traffic, no traffic was set to 0, mild was set to 1, and every other level was set to 2.

4. Results

Model	Class	Precision	Recall	F1 score	Overall Accuracy
Crosswalk detection	Class 0	0.70	0.97	0.80	0.81
	Class 1	0.97	0.70	0.81	
Crosswalk sign detection	Class 0	0.98	0.70	0.81	0.85
	Class 1	0.77	0.98	0.87	
Sidewalk detection	Class 0	0.50	0.28	0.36	0.54
	Class 1	0.55	0.46	0.50	
	Class 2	0.55	0.91	0.69	
Stop sign detection	Class 0	0.84	0.94	0.89	0.89
	Class 1	0.94	0.85	0.89	
Traffic lights detection	Class 0	0.37	0.81	0.51	0.54
	Class 1	0.84	0.42	0.56	

Table 1: Overall results of all the models used

The models were tested on the street view images from a 1-mile radius of the Indiana University, Bloomington campus. The view images were downloaded using Google street view API. The result of the safety score is shown in Figure 3, and the front end is shown on Figure 4. You can see that while there are reasonable predictions, but there are also some unpredicted results.

Table 1 shows the confusion metrices. Though some of the accuracies do show low values, it was compensated by taking the max probability of an object from the 4 images of each coordinate.

Figure 3 indicates some good results and bad results of the safety score algorithm. For example, Figure 3-(c) is correctly classified as having two sidewalks, where as a building wall in Figure 3-(i) is also classified as two sidewalks.

Figure 4 are two examples of the paths you can get using the front end. The paths do show a different path than direct,

Figure 3 : one of the images belonging to some coordinates and the safety scores. Top row are true positives, and the bottom row are incorrect examples. Starting from the left, cross walk, cross signs, sidewalks, traffic lights, stop signs examples.

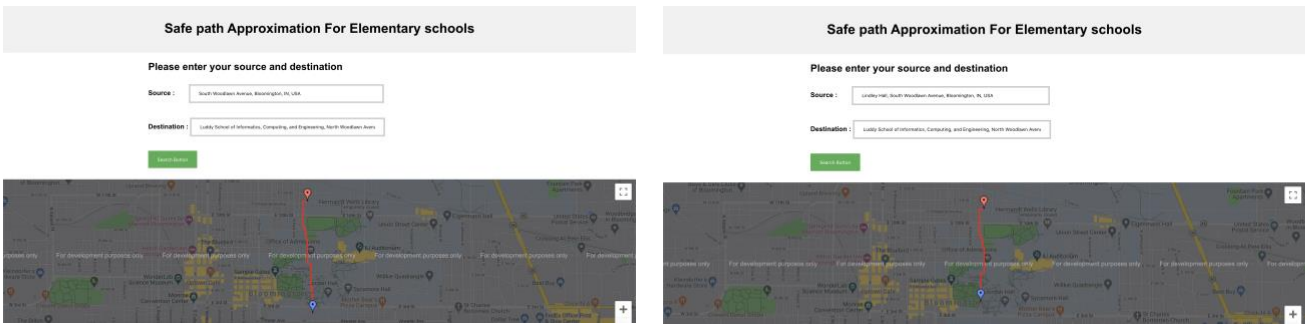

Figure 4 : Two examples of paths given by the front end. It does give some different path then direct path but is failing to distinguish roads.

which is natural given that the heuristic covers safety and traffic of the coordinate. However, it fails to distinguish roads, and shows a path going through the building.

Our implementation of the project was written in python, and the models were written using tensorflow keras framework[14]. All the model trainings were done on the Carbonate deep learning node from IU's Red system. The GPU used was Tesla P100.

5. Discussion

Another factor is that since there were no labeled data on the number of sidewalks in a street view image, the images had to be self-labeled. This is a very error-prone highly biased approach. The model would be improved if given more data and more people labeling them, which is an approach done by the project sidewalk.[2]

For detecting crosswalks, we tried both the state-of-the-art Computer Vision techniques and Deep Learning methods. Using Computer Vision techniques finding edges and contours and group contours together to detect crosswalk. This method failed mainly due to the wide range of images and it was not possible to set the parameters for the same. We also tried using ORB descriptors for detecting crosswalk signs in the images. Mainly due to the size constraints this method failed.

The training of the crosswalk model can be improved. This model was trained by taking screenshots from Google street views around Bloomington and augmented the same. We also used data from IARA and GOPRO [13] where they had little training dataset available for crosswalks. Since this was out of the domain dataset, this did not help a lot in classifying crosswalks for Bloomington data. Since the training set was less, this model had difficulties in classifying crosswalks which are far in the image and had occlusions.

Safety score algorithm and the heuristic function's hyperparameters, like weights and the way the scores are calculated from each model was set by intuition. Some practicing by tweaking the parameters will improve the path given by the project.

Especially, the current overall safety score algorithm is a relatively simple equation. Another way of evaluating suggested by the team was by adding weights to the coordinates based on the Euclidean distance from the school since the roads should be safer if it is closer to the school grounds.

The shortest path, though giving some reasonable results, do not give the correct path. One of the major reasons seems to be failing to distinguish coordinates defined as roads. This seems to be a result of safety scores indicating a building safe to walk and the fact that there are no methods preventing from using the building as a path. Nevertheless, it still shows that with the right amount of weight and value, it would be possible to alter the distance wise shortest path to safety wise better path.

The overall models used are simple AlexNet[5] and ResNet[6]. There are multiple state of the art object image classification algorithms that can be used, but it was our understanding that since this task is a simple binary or 3 type classification job, a complicated model would just overfit to the training data. Thus, there was more concentration on using the results for the project.

6. Conclusion and Future work

The project aims to give the safety score of a school and provide an alternative walking path from source to destination around the school. We used simple CNN models with street view images as input to calculate the safety score of each coordinate. Though there were good results, there were also some critical errors and observations that could be improved in future work.

The training of the sidewalk model has some improvements that can be made. It is mainly trained on around Templeton elementary school and University elementary school. Towns around elementary schools tend to have more variant sidewalks, like shape, location, and color. This could be a reason the model failed to find quite a few sidewalks around the IU campus, where it has relatively planned, fixed sidewalks.

Given that there are many common features between crosswalks and no crosswalks, more training data is required to capture these limitations. Also, these images are trained only for day datasets, it can also be extended to include night images as well.

The major update would be conducted on finding the shortest path, possibly adding some advanced path finding features used by other applications.

7. References

[1] https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812383

[2] Lam, Tiffany. "Too far to walk?" molly green's blog, Safe Routes Partnership

[3] https://sidewalk-sea.cs.washington.edu/

[4] http://www.walkitarizona.org/

[5] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems. 2012.

[6] He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[7] Ioffe, Sergey, and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." arXiv preprint arXiv:1502.03167 (2015).

[8] Nesterov, Yurii. "A method for unconstrained convex minimization problem with the rate of convergence O (1/k^ 2)." Doklady an ussr. Vol. 269. 1983.

[9] Qian, Ning. "On the momentum term in gradient descent learning algorithms." Neural networks 12.1 (1999): 145-151.

[10] Kingma, D. P., & Ba, J. L. (2015). Adam: a Method for Stochastic Optimization. International Conference on Learning Representations, 1–13.

[11] https://pypi.org/project/PyGeodesy/

[12] https://docs.microsoft.com/en-us/bingmaps/rest-services/routes/calculate-a-route

[13] Rodrigo F Berriel, Franco Schmidt Rossi, Alberto F. de Souza, and Thiago Oliveira-Santos. Automatic Large-Scale Data Acquisition via Crowdsourcing for Crosswalk Classification: A Deep Learning Approach

[14] Abadi, Martín, et al. "Tensorflow: A system for large-scale machine learning." 12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16). 2016.