TRAILBot Real-Time Trail Segmentation

Andres Cervera Rozo, Mehtab Malik

***Abstract*—The Toronto Robotics and Artificial Intelligence Laboratory (TRAIL) has been developing an autonomous robot, Trailbot, that is capable of traversing trails. Trailbot uses a Convolutional Neural Network (CNN) to semantically segment incoming real-time video feed and identify the trail to follow. The current CNN implementation is limited by its dataset and outdated architecture. This project aims to improve the Trail Detection CNN's runtime speed, robustness, and accuracy by expanding the dataset, augmenting the data, adjusting the architecture, and tuning the hyperparameters of the adjusted architecture. The new CNN implementation is compared with the current CNN implementation and is found to be more robust and accurate for predicting all common trail types.**

I. Introduction

The team at the Toronto Robotics and Artificial Intelligence Laboratory (TRAIL) has been developing an autonomous robot, the Trailbot, to traverse through varying hiking trails, identify the different paths, and offer snacks to the passing hikers. In order to ensure that it is capable of staying within the boundaries of the trail, the robot must be able to perceive where the trail is, and act accordingly.

The solution discussed in this paper details a semantic segmentation approach to this problem whereby a Convolutional Neural Network (CNN) was trained to classify each pixel as either being a member of the 'trail' or 'non-trail' class. This trained model will then be transferred to the robot such that it is able to process the frames of the incoming real-time video feed and identify the trail to follow. As such, the model that is utilized must be appropriate for a mobile robotic platform where computational power is finite and there are many other computations being performed in parallel. Because this robot will be operating in an unsupervised, open-world environment, it is imperative that the model is adept at performing in varying types of trails and environmental conditions that may appear.

II. Motivation and Related Work

Earlier this year, significant work was undertaken to enhance Trailbot's capability to segment camera frames and detect pixels associated with a trail. The progress made on this project before September serves as a benchmark for any subsequent developments.

The current model, PSPNet with a ResNet-50 backbone for semantic segmentation, is deemed suitable for the task. However, it is worth noting that this model, developed in 2016, is no longer considered state-of-the-art compared to more recent models.

The dataset used to train the model was exclusively collected by lab members and comprised approximately 1000 images from various points along the same dirt trail near the University of Toronto Institute for Aerospace Studies (UTIAS) campus. These images were acquired using the onboard camera of the robot, resulting in images of seemingly low resolution quality with a slight green tint. Conversations with a previous contributor to the project revealed that when evaluating the benchmark model's performance using the mean Intersection of Union (mIoU), the results were inconsistent and appeared to be overfit to the specific dirt trail from which the data was sourced.

In light of comprehending the capabilities and limitations of the existing solution, the project's key objectives were formulated. The viability of potential solutions was assessed based on their adherence to the following objectives: suitability for online semantic segmentation of live incoming data; adaptability to varying environmental conditions and types of trails; and accuracy in predicting correct trail paths. The first objective was evaluated through research into the requisite computational requirements, while the latter two objectives were assessed using both mIoU and pixel accuracy metrics.

III. Methodology

This project aims to enhance Trailbot's trail detection process by actively focusing on the three defined key objectives: speed, robustness, and accuracy. To achieve these objectives, we employed the following four-step approach:

1. **Dataset Expansion**
2. **Data Augmentation**
3. **Architecture Adjustment**
4. **Hyperparameter Tuning**

A. Dataset Expansion

The current Trailbot dataset, comprising 1000 images, is restricted to dirt trails, leading to significant limitations in accurately identifying other common trail types such as gravel, asphalt, and concrete.

To enhance the model's robustness and accuracy, the Trailbot dataset was expanded by incorporating 400 images. This supplementary dataset, gathered and segmented by the team, encompasses various asphalt, concrete, dirt, and gravel trails around Toronto.

B. Data Augmentation

Despite the inclusion of the additional 400 images, the Trailbot dataset remains relatively small. Training a model on a limited dataset can result in overfitting, wherein the model may not only learn the underlying patterns, but also the noise or specific characteristics unique to the restricted dataset.

To mitigate overfitting and enhance the model's robustness, four data augmentation techniques were applied randomly to the training images at every epoch:

- **Mirror:** Flip the image along its vertical axis.
- **Scale:** Resize the image while preserving its aspect ratio.
- **Crop**: Extract a portion of the image.
- **Gaussian Blur:** Blur the image to reduce noise and suppress detail.

C. *Architecture Adjustment*

The existing Trailbot model is composed of a ResNet-50 backbone and the PSPNet method. As discussed earlier, this architecture from 2016 is no longer considered state-of-the-art due to substantial developments in the field of semantic segmentation in recent years[1].

To better align with the objectives of the project, an exploration of state-of-the-art semantic segmentation models was conducted to determine the most appropriate combinations of backbones and methods. The architectures were evaluated on the following criteria:

- **Performance:** How well does the architecture perform on segmentation tasks using existing popular datasets (e.g., Cityscapes or COCO)?
- **Runtime:** Is the architecture suitable for real-time applications on mobile robots?
- **Computatiation Requirement:** How many weights need to be optimized and is it feasible with the number of available GPUs?
- **PyTorch Compatibility:** How complex is the integration of these models with PyTorch?

The expanded dataset was partitioned into 60%, 20%, 20% for training, validating, and testing, respectively. These partitions were then used for training, validating, and testing each architecture to ensure reliable performance results for comparison. The architecture performances were evaluated using the following metrics:

- **Pixel Accuracy:** The percentage of accurately classified pixels in the image.
- **Mean Intersection over Union (mIoU):** The percentage of overlap between the predicted segmentation and the ground-truth segmentation.

The architecture's performance was computed as the average of its pixel accuracy and mIoU. The architecture demonstrating the highest performance was then selected as the new architecture for the Trailbot Trail Detection CNN.

D. *Hyperparameter Tuning*

Hyperparameters can have a significant impact on a model's robustness, speed, and accuracy.

To optimize the new Trailbot Trail Detection CNN, the following hyperparameters were tuned:

- **Batch Size:** The number of samples processed by the model in one iteration.
- **Epochs:** The number of passes of the entire dataset through the model during training.
- **Learning Rate:** The rate at which the model parameters are updated during training.

IV. Results and Discussion

A. *Dataset Expansion*

To assess the impact of dataset expansion, three rounds of training and testing were carried out on the original PSPNet + ResNet-50 architecture. Subsequently, the resultant models underwent evaluation based on pixel accuracy and mIoU metrics. The outcomes of these iterations are illustrated in Fig. 1.

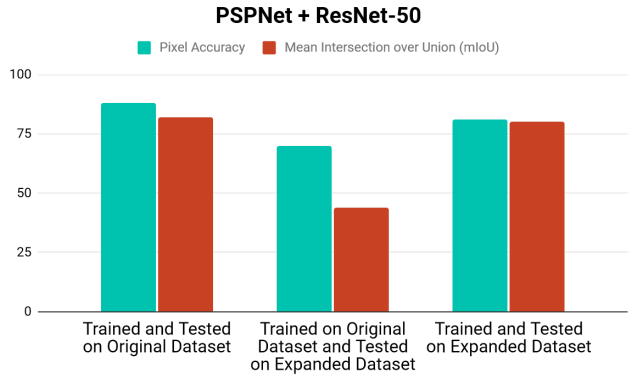

Figure 1. PSPNet + ResNet-50 Architecture Performance on Various Training and Testing Datasets

The first iteration involved training and testing the model exclusively using the original Trailbot dataset. From this iteration, it was concluded that the original model is proficient in detecting dirt trails.

The second iteration entailed training the model solely with the original Trailbot dataset and testing it using data from the expanded dataset. This iteration revealed that, as hypothesized, due to the limitations of the original dataset, the model performed poorly when attempting to detect other common trail types such as gravel, asphalt, and concrete.

The final iteration involved training and testing the model using the expanded dataset. Conclusively, it was determined that the expanded dataset significantly improved the model's ability to detect most common trails.

A comparison between the second and third iterations revealed a 23.5% increase in the model's performance when trained using the expanded dataset and tested on all common trail types (gravel, asphalt, dirt, and concrete). To ensure the validity of the results, the test dataset remained consistent for both the second and third iterations.

A visual comparison of the predictions from the second and third iteration models is presented in Fig. 2.

Figures 1 and 2 clearly illustrate that the old model, trained solely on the original dataset, exhibited signs of overfitting, leading to difficulties in predicting non-dirt type trails. In contrast, predictions from the original model, trained on the expanded dataset, demonstrate increased robustness and accuracy.

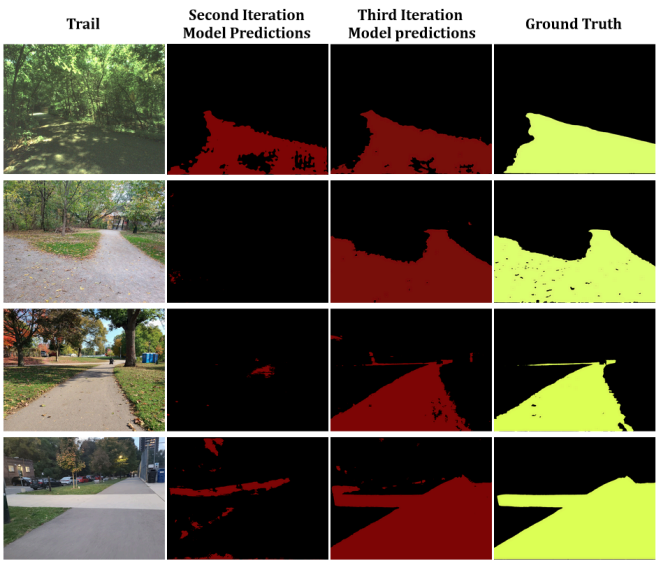

Figure 2. Visual Comparison of the Predictions of the Old Model Trained on the Original Dataset vs. Expanded Dataset

B. *Architecture Selection*

As discussed in Section III.C, the selection of architectures was guided by specific criteria, encompassing aspects such as performance, runtime, computation requirement, and compatibility with PyTorch.

Based on the reported mIoU scores of state-of-the-art models in real-time semantic segmentation on the Cityscapes dataset, the following models were chosen[2]:

- **LEDNet + ResNet-50**
- **DeepLabV3 + ResNet-50**
- **BiSeNet + ResNet-18**
- **ICNet + ResNet-50**

These particular models also demonstrated compatibility with PyTorch and were found to be adept for online performance with a single GPU[3][4][5][6].

C. *Architecture Evaluation*

To assess the performance of the selected architectures, consistency was maintained in both the dataset and hyperparameters. The specific hyperparameter values used for the evaluation are detailed in Table I, while the performance metrics, including those for the current PSPNet + ResNet-50 architecture, are presented in Table II for a comprehensive benchmark comparison.

TABLE I. ARCHITECTURE EVALUATION HYPERPARAMETER VALUES

Hyperparameter	Value
Learning Rate	0.003
Batch size	8
Epochs	10

TABLE II. ARCHITECTURE EVALUATION PERFORMANCE RESULTS

Architecture	Pixel Accuracy (%)	mIoU (%)	Performance (%)
LEDNet + ResNet-50	92.049	83.525	87.787
PSPNet + ResNet-50	89.758	83.478	86.618
DeepLabV3 + ResNet-50	87.738	82.870	85.304
BiSeNet + ResNet-18	87.865	80.320	84.093
ICNet + ResNet-50	82.613	82.234	82.234

From Table II, it can be seen that the architecture with the ResNet-50 backbone and LEDNet method had the best results in terms of pixel accuracy, mIoU, and overall performance. It outperformed the original PSPNet + ResNet-50 architecture by 1.1% in overall performance. This was expected because LEDNet, a Lightweight Encoder-Decoder Network, is a state-of-the-art model specifically designed for semantic segmentation tasks with relatively low available computational power[3].

D. *Hyperparameter Tuning*

To optimize the LEDNet + ResNet-50 model, the learning rate, batch size, and epochs were tuned and then evaluated using the pixel accuracy and mIoU metrics. The specific values tested for each hyperparameter are detailed in Table III, with the tuning results provided in Table IV.

TABLE III. HYPERPARAMETER TUNING VALUES

Hyperparameter	Tuning Values
Learning Rate	0.002, 0.003, 0.004, 0.005, 0.006
Batch size	2, 4, 8, 16, 32
Epochs	10, 20, 30, 40, 50, 100

TABLE IV. HYPERPARAMETER TUNING PERFORMANCE RESULTS

Before Tuning			
Hyperparameter			**Performance (%)**
Learning Rate	Batch Size	Epochs	
0.003	8	10	87.787
After Tuning			
Hyperparameter			**Performance (%)**
Learning Rate	Batch Size	Epochs	
0.004	4	30	89.379

From Table IV, it can be seen that the tuned hyperparameter values for the LEDNet + ResNet-50 model were:

- **Learning Rate:** 0.004
- **Batch Size:** 4
- **Epochs:** 30

The tuned model demonstrated a performance of 89.379%, reflecting a 1.592% increase in performance as a result of the tuning process.

V. Future Works

Moving forward, the progress achieved in this project will be adopted and integrated by the Trail Lab team to enhance the performance of the Trailbot. Consequently, there remains ample opportunity for improvements to ensure the robot can operate more robustly in diverse trail conditions.

The selection of the model itself has been made from among other similar state-of-the-art models. Furthermore, the associated hyperparameters have undergone tuning. While there may be other future model options with marginally better results, the emphasis of forthcoming work should shift away from the neural network architecture.

The primary limitation affecting the model's robustness is the scarcity of training data for various conditions. Despite significant strides in this area, there is still considerable room for improvement. In addition to acquiring more images depicting different trails and diverse environmental conditions, exploring the feasibility of artificial data generation using autoencoders could be advantageous. This technique, increasingly popular in the industry, has the potential to significantly enhance the dataset size. For the expanded dataset, further data augmentation techniques may still be explored, such as varying illumination, fog, and introducing artificial motion blur. These techniques aim to better simulate the potential scenarios that the moving Trailbot may encounter.

Another technique to enhance the final results of the segmentation mask involves mitigating noise. Post-processing techniques, such as leveraging OpenCV to identify contours in images and preserving blobs above a certain size, may help eliminate noise caused by small misclassified non-trail objects. Additionally, post-processing could involve isolating and removing noise in masks that only appear for a singular frame at a time, smoothing the transient results of the trail boundaries.

The third area for improvement involves training the model to perform multi-class semantic segmentation. Instead of the binary classification of pixels as either 'trail' or 'non-trail,' the Trailbot could potentially identify multiple classes. This might include determining the specific type of trail, providing valuable information for the navigation stack when choosing between multiple trails. Additionally, as a next step, it could be beneficial for the robot to identify both static and dynamic obstacles along the trail. These obstacles could encompass hikers, pylons, dogs, bikes, and more. By enhancing the semantic segmentation model to accurately identify these distinct objects, the navigation stack would be better equipped to make informed decisions.

VI. Conclusion

The current Trailbot Trail Detection CNN has limitations regarding its speed, robustness, and accuracy due to its limited dataset and outdated architecture. In the report, a four-step systematic approach is applied to improve the current Trailbot Trail Detection CNN. First, the original dataset is expanded with 400 images of asphalt, dirt, gravel, and concrete trails across Toronto. Then, the data is augmented using mirroring, scaling, cropping, and Gaussian blurring techniques. Next, several state-of-the-art semantic segmentation architectures are selected and evaluated. Finally, the hyperparameters are tuned for the highest performing architecture. From this approach, several conclusions can be made:

- The original model trained on the original dataset is overfit and therefore struggles to predict non-dirt type trails.
- The use of the expanded dataset for training models results in more accurate predictions across all trail types compared to the original dataset.
- The LEDNet + ResNet-50 is the most suitable architecture for the Trailbot Trail Detection CNN.
- The LEDNet + ResNet-50 architecture outperforms the PSPNet + ResNet-50 architecture by 1.1% in overall performance before tuning.
- The tuned hyperparameter values for the LEDNet + ResNet-50 architecture are a learning rate of 0.004, batch size of 4, and epochs of 30.
- The tuned LEDNet + ResNet-50 architecture has a 89.379% performance accuracy based on pixel accuracy and mIoU metrics.

The Trailbot Trail Detection CNN still has numerous areas of improvement. Next steps would include further expanding the dataset, implementing more data augmenting techniques (such as illumination, fog, and motion blurring), implementing post-processing techniques, and expanding from a binary to a multi-class semantic segmentation.

Acknowledgment

We would like to acknowledge and thank Zhaodong (Frank) Jiang for his work in developing the semantic segmentation model that was used as a benchmark for comparison. Additionally, his insight into the different areas for improvement was invaluable in defining the objectives of the project. Furthermore, this project would not have been possible without the work of the TRAIL team in developing the Trailbot itself.

References

[1] "Papers with code - pyramid scene parsing network," Pyramid Scene Parsing Network | Papers With Code, https://paperswithcode.com/paper/pyramid-scene-parsing-network (accessed Dec. 15, 2023).

[2] “Papers with code - cityscapes test benchmark (real-time semantic segmentation),” The latest in Machine Learning, https://paperswithcode.com/sota/real-time-semantic-segmentation-on-cityscapes (accessed Dec. 15, 2023).

[3] “Papers with code - lednet: A lightweight encoder-decoder network for real-time semantic segmentation,” LEDNet: A Lightweight Encoder-Decoder Network for Real-Time Semantic Segmentation | Papers With Code, https://paperswithcode.com/paper/lednet-a-lightweight-encoder-decoder-network (accessed Dec. 15, 2023).

[4] “Papers with code - deeplabv3 explained,” Explained | Papers With Code, https://paperswithcode.com/method/deeplabv3 (accessed Dec. 15, 2023).

[5] “Papers with code - bisenet: Bilateral segmentation network for real-time semantic segmentation,” BiSeNet: Bilateral Segmentation Network for Real-time Semantic Segmentation | Papers With Code, https://paperswithcode.com/paper/bisenet-bilateral-segmentation-network-for (accessed Dec. 15, 2023).

[6] “Papers with code - icnet for real-time semantic segmentation on high-resolution images,” ICNet for Real-Time Semantic Segmentation on High-Resolution Images | Papers With Code, https://paperswithcode.com/paper/icnet-for-real-time-semantic-segmentation-on (accessed Dec. 15, 2023).