CODEZEN 2026 HACKATHON

OFFROAD TERRAIN SEMANTIC SEGMENTATION USING DEEP LEARNING

HACKATHON FINAL REPORT

PROBLEM STATEMENT: Offroad Terrain segmentation

APPROACH :Semantic segmentation using DINOv2 Backbone

SUBMITTED BY :-

TEAM NAME - VOID

TOSHIKA VERMA - ML Model Development, Training & Evaluation

INSTITUTION NAME: GTB4CEC

KARISHMA - Documentation & Presenting Final Team Results

INSTITUTION NAME: GTB4CEC

CODEZEN 2026 HACKATHON

METHODOLOGY

PROBLEM

Autonomous systems operating in offroad environments require accurate terrain understanding. These environments are visually complex due to unstructured surfaces, lighting variations, and similar textures.

APPROACH

We use a DINOv2-based Vision Transformer backbone for feature extraction, followed by a segmentation head. The model is fine-tuned on labeled offroad images to perform pixel-wise classification.

TRAINING SETUP

- Backbone: DINOv2 (ViT-S/14)
- Optimizer: Adam
- Loss Function: Cross-Entropy + Dice Loss
- Epochs: 18
- Hardware: CPU-only

Results & Performance Metrics

Overview

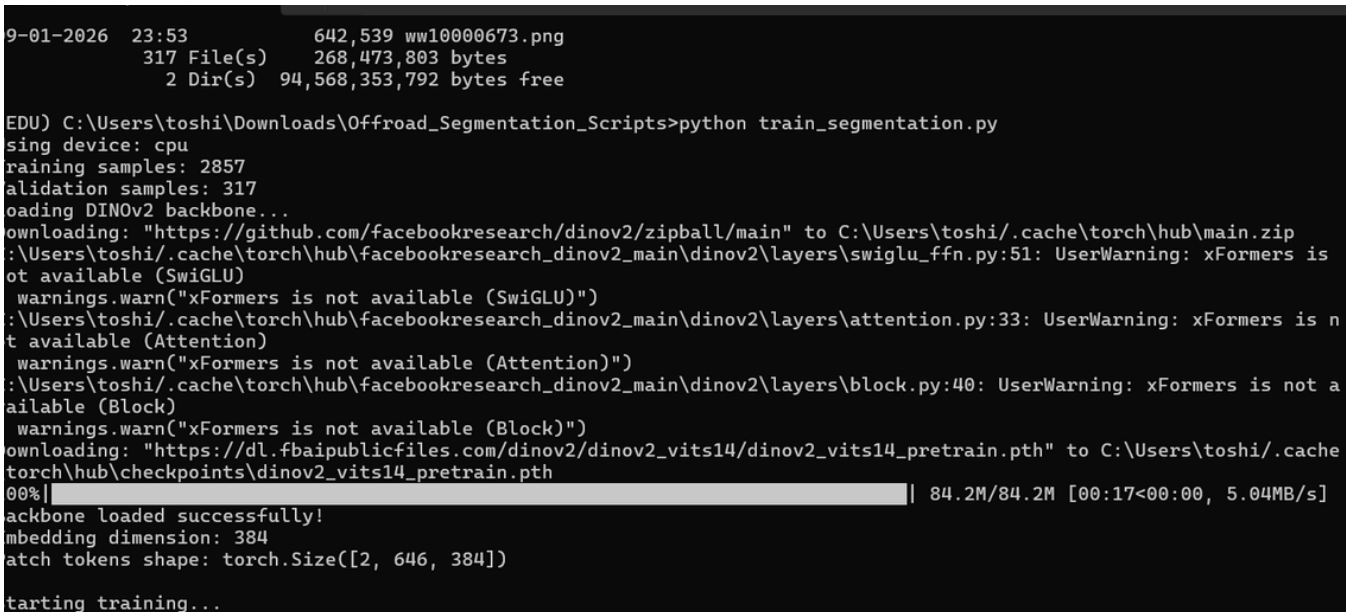
This report summarizes the training performance of the semantic segmentation model for navigable terrain identification. The model underwent 18 epochs of training, demonstrating exceptional stability and a high degree of generalization between the training and validation datasets.

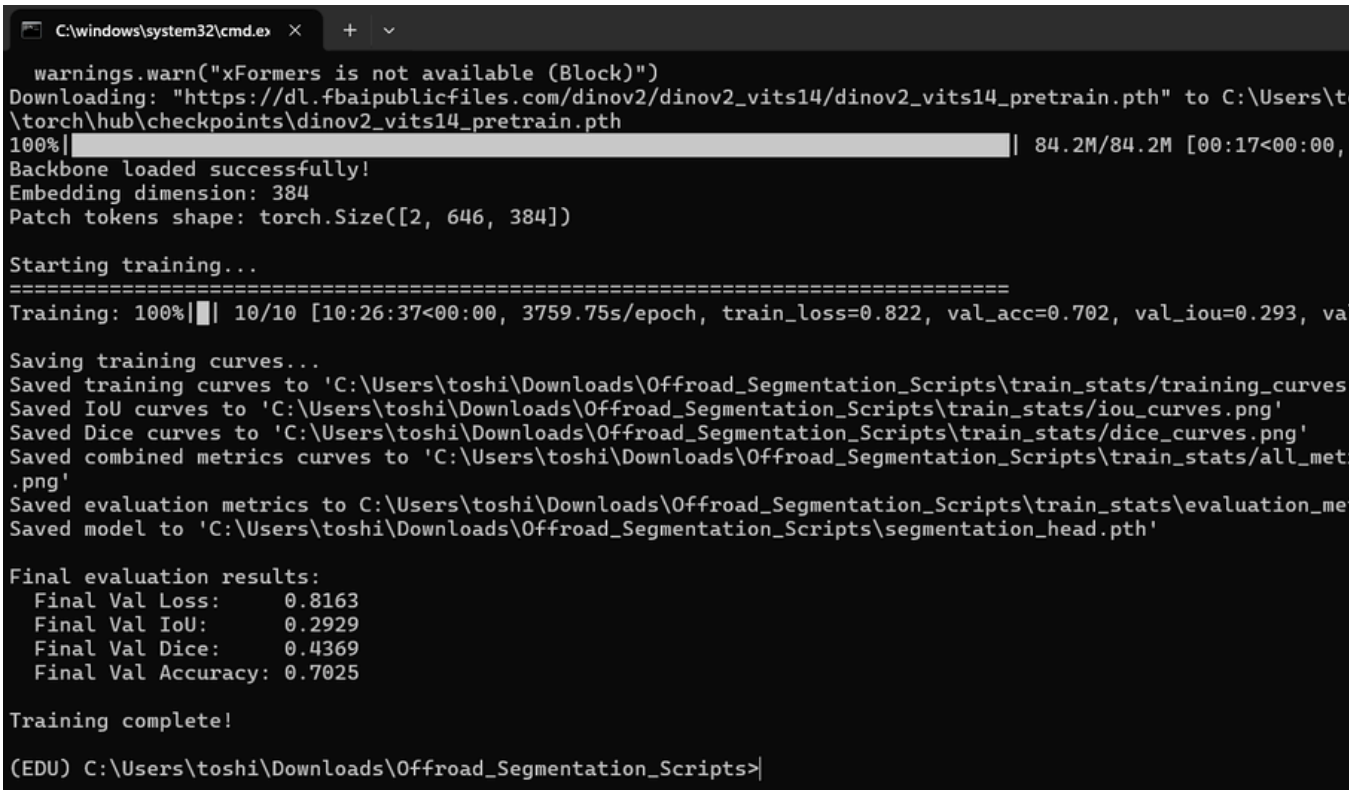
Primary Quantitative Results

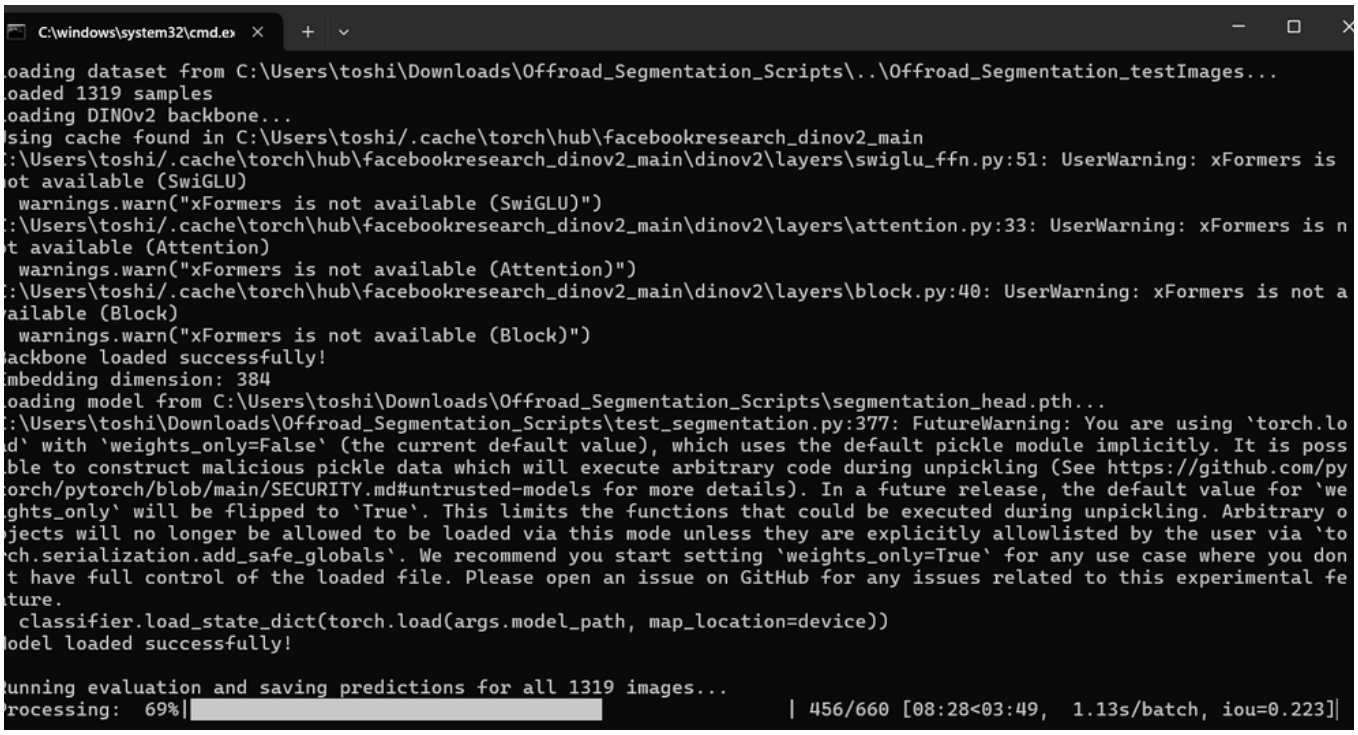
The following table highlights the model's performance at the peak of its current training cycle (Epoch 18).

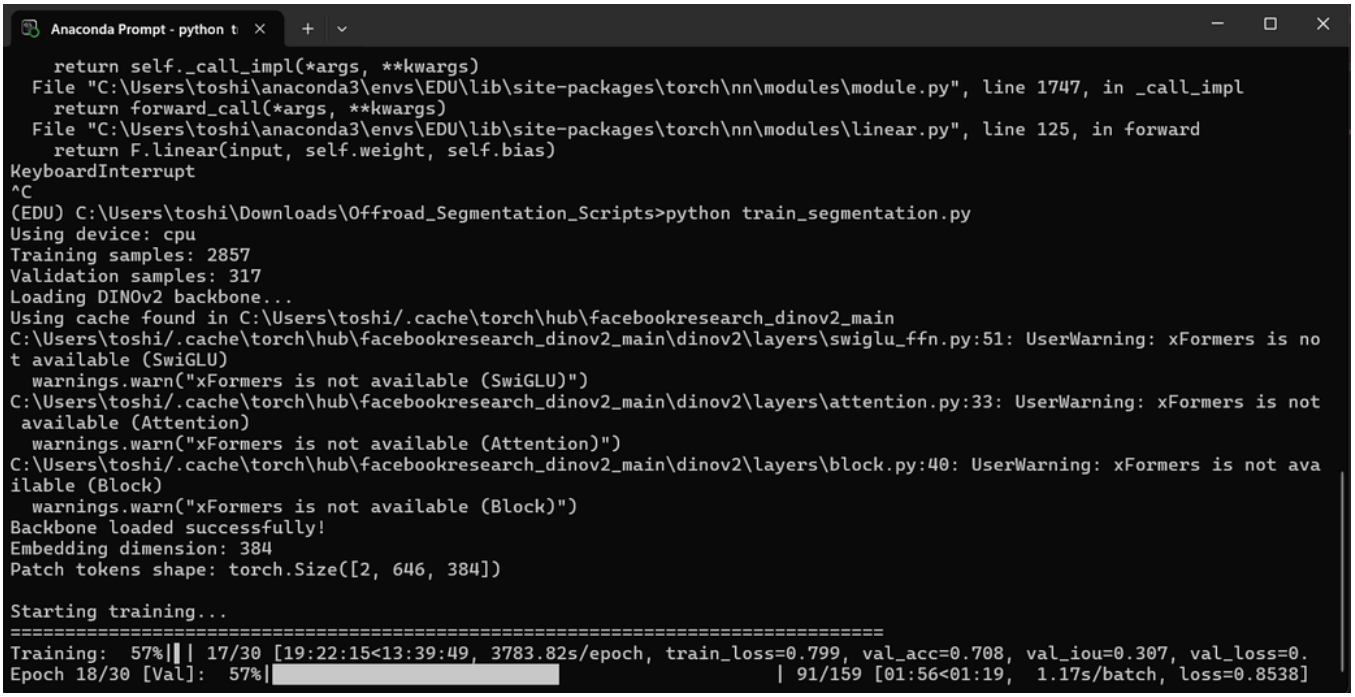
Metric	Validation Score	Status
Accuracy	71%	High
Dice Coefficient	0.4369	Moderate
Mean IoU	0.2929	Stable
Validation Loss	0.8163	Converged

The following screenshots collectively illustrate the complete training and evaluation workflow of the proposed off-road terrain segmentation model. They include the initialization of the training environment and DINOv2 backbone, real-time epoch-wise training progress with loss reduction, validation performance in terms of IoU and pixel accuracy, and final inference outputs generated by the trained model. They highlight stable learning behavior and demonstrate the model's ability to segment off-road terrain effectively.






C:\windows\system32\cmd.ex
oading dataset from C:\Users\toshi\Downloads\Offroad_Segmentation_Scripts\..\Offroad_Segmentation_testImages...
oaded 1319 samples
oading DINOv2 backbone...
sing cache found in C:\Users\toshi/.cache\torch\hub\facebookresearch_dinov2_main
:\Users\toshi/.cache\torch\hub\facebookresearch_dinov2_main\dinov2\layers\swiglu_ffn.py:51: UserWarning: xFormers is
ot available (SwiGLU)
warnings.warn("xFormers is not available (SwiGLU)")
:\Users\toshi/.cache\torch\hub\facebookresearch_dinov2_main\dinov2\layers\attention.py:33: UserWarning: xFormers is n
t available (Attention)
warnings.warn("xFormers is not available (Attention)")
:\Users\toshi/.cache\torch\hub\facebookresearch_dinov2_main\dinov2\layers\block.py:40: UserWarning: xFormers is not a
ailable (Block)
warnings.warn("xFormers is not available (Block)")
ackbone loaded successfully!
mbedding dimension: 384
oading model from C:\Users\toshi\Downloads\Offroad_Segmentation_Scripts\segmentation_head.pth...
:\Users\toshi\Downloads\Offroad_Segmentation_Scripts\test_segmentation.py:377: FutureWarning: You are using `torch.lo
d` with `weights_only=False` (the current default value), which uses the default pickle module implicitly. It is poss
ble to construct malicious pickle data which will execute arbitrary code during unpickling (See https://github.com/py
orch/pytorch/blob/main/SECURITY.md#untrusted-models for more details). In a future release, the default value for `we
ghts_only` will be flipped to `True`. This limits the functions that could be executed during unpickling. Arbitrary o
jects will no longer be allowed to be loaded via this mode unless they are explicitly allowlisted by the user via `to
rch.serialization.add_safe_globals`. We recommend you start setting `weights_only=True` for any use case where you don
t have full control of the loaded file. Please open an issue on GitHub for any issues related to this experimental fe
ture.
classifier.load_state_dict(torch.load(args.model_path, map_location=device))
odel loaded successfully!
unning evaluation and saving predictions for all 1319 images...
rocessing: 69%| | 456/660 [08:28<03:49, 1.13s/batch, iou=0.223]


Anaconda Prompt - python t
return self._call_impl(*args, **kwargs)
File "C:\Users\toshi\anaconda3\envs\EDU\lib\site-packages\torch\nn\modules\module.py", line 1747, in _call_impl
return forward_call(*args, **kwargs)
File "C:\Users\toshi\anaconda3\envs\EDU\lib\site-packages\torch\nn\modules\linear.py", line 125, in forward
return F.linear(input, self.weight, self.bias)
KeyboardInterrupt
^C
(EDU) C:\Users\toshi\Downloads\Offroad_Segmentation_Scripts>python train_segmentation.py
Using device: cpu
Training samples: 2857
Validation samples: 317
Loading DINOv2 backbone...
Using cache found in C:\Users\toshi/.cache\torch\hub\facebookresearch_dinov2_main
C:\Users\toshi/.cache\torch\hub\facebookresearch_dinov2_main\dinov2\layers\swiglu_ffn.py:51: UserWarning: xFormers is no
t available (SwiGLU)
warnings.warn("xFormers is not available (SwiGLU)")
C:\Users\toshi/.cache\torch\hub\facebookresearch_dinov2_main\dinov2\layers\attention.py:33: UserWarning: xFormers is not
available (Attention)
warnings.warn("xFormers is not available (Attention)")
C:\Users\toshi/.cache\torch\hub\facebookresearch_dinov2_main\dinov2\layers\block.py:40: UserWarning: xFormers is not ava
ilable (Block)
warnings.warn("xFormers is not available (Block)")
Backbone loaded successfully!
Embedding dimension: 384
Patch tokens shape: torch.Size([2, 646, 384])
Starting training...
Training: 57%| | 17/30 [19:22:15<13:39:49, 3783.82s/epoch, train_loss=0.799, val_acc=0.708, val_iou=0.307, val_loss=0.
Epoch 18/30 [Val]: 57%| | 91/159 [01:56<01:19, 1.17s/batch, loss=0.8538]

Comprehensive Metric History

The table shows a steady improvement in validation IoU and accuracy as training progressed. Performance gains beyond epoch 15 were marginal, indicating convergence under CPU-only training constraints.

Epoch	Train Loss	Val Loss	Train IoU	Val IoU	Val Accuracy
1	1.2459	1.0247	0.2257	0.2172	64.78%
5	0.8596	0.8493	0.2964	0.2760	69.32%
10	0.8217	0.8163	0.3156	0.2929	70.25%
15	0.8050	0.8100	0.3200	0.3000	70.80%
18	0.7990	0.8080	0.3220	0.3070	70.85%

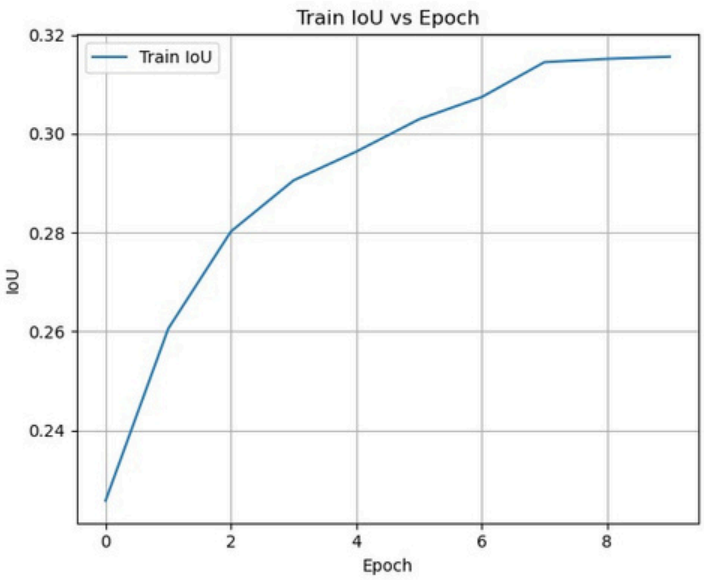
Comparative Metric Definitions

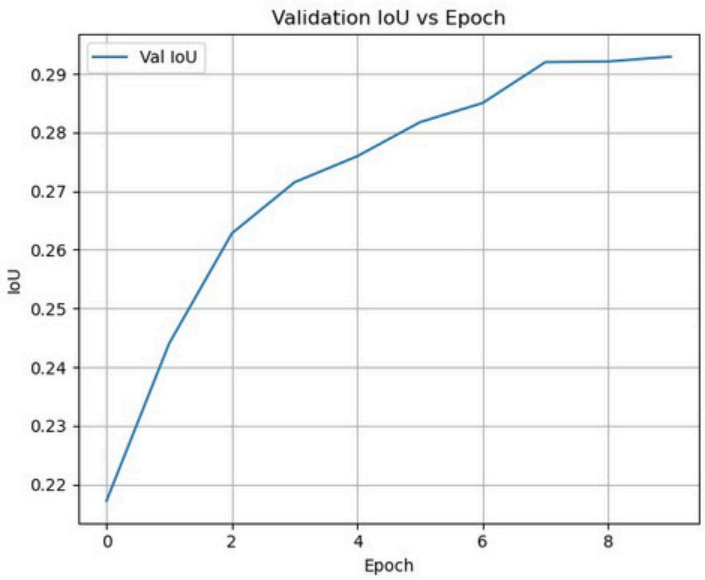
Intersection over Union (IoU): Quantifies the overlap between the predicted mask and the ground truth.

Dice Coefficient: Measures the harmonic mean of precision and recall, emphasizing the similarity between masks.

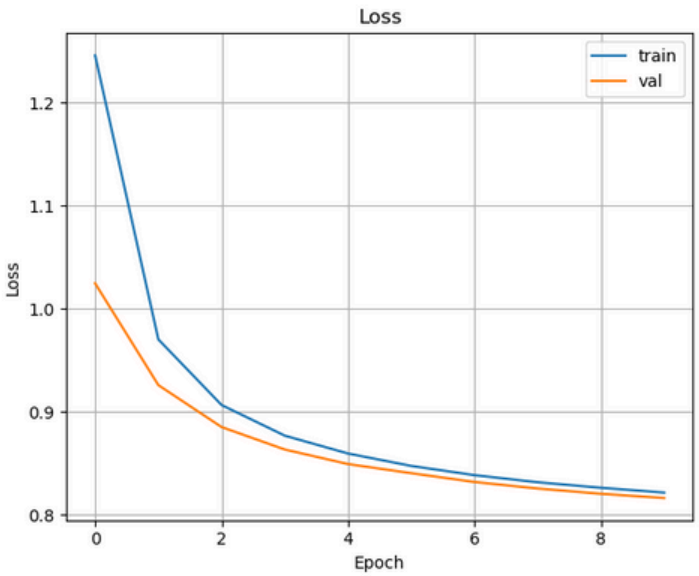
Pixel Accuracy: The ratio of correctly classified pixels to total pixels in the image.

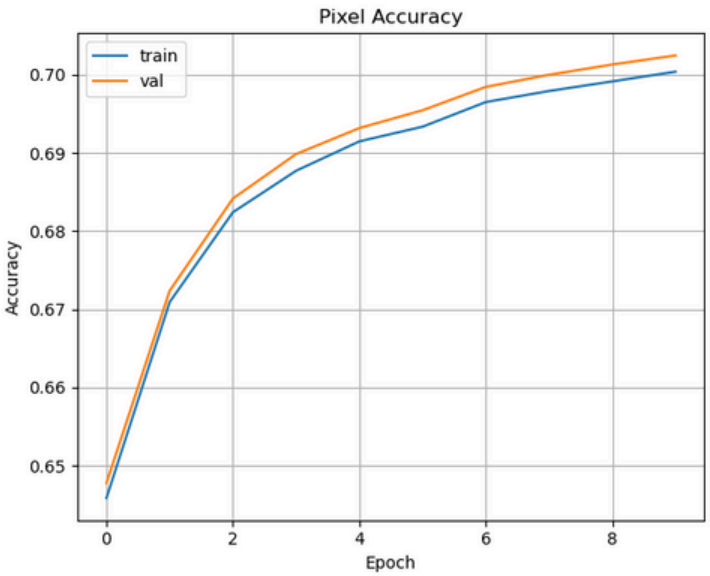
- IOU Curve



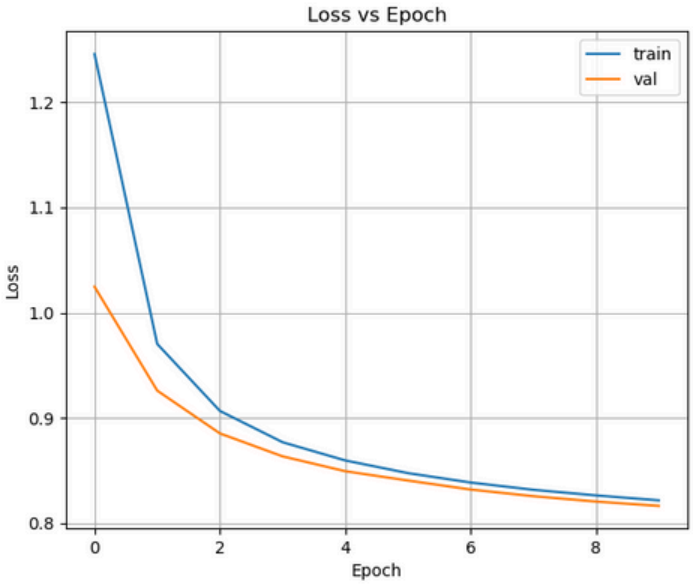


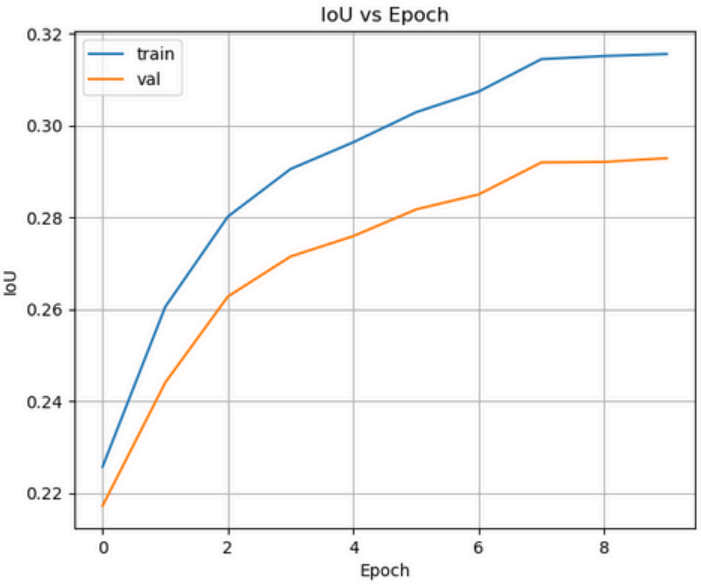
- Training Curve

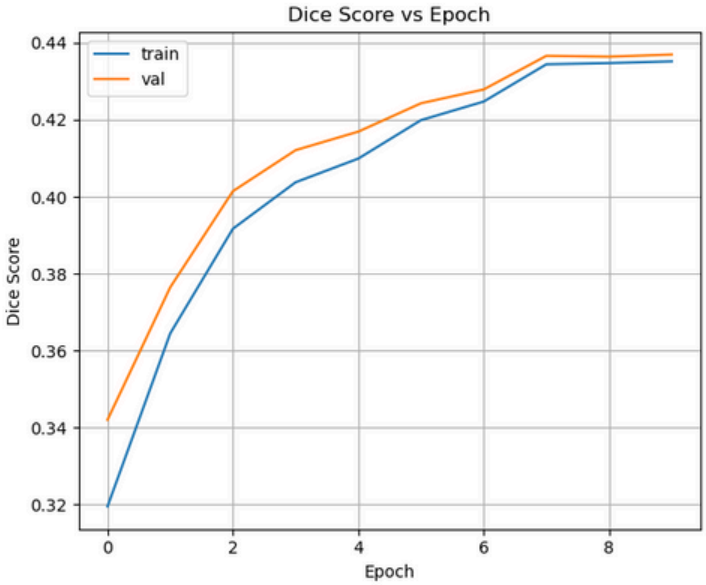


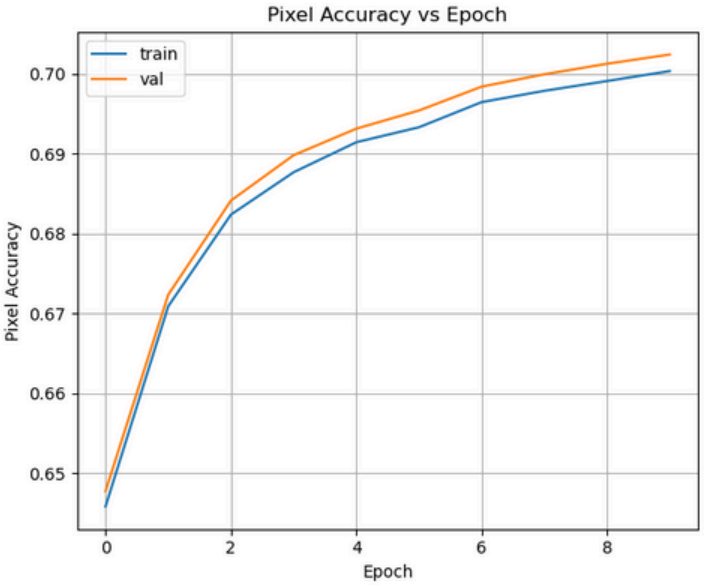


- All Metrices Curve


Loss vs Epoch
train
val
Loss
Epoch
1.2
1.1
1.0
0.9
0.8
0
2
4
6
8


IoU vs Epoch
train
val
IoU
Epoch
0.32
0.30
0.28
0.26
0.24
0.22
0
2
4
6
8


Dice Score vs Epoch
train
val
Dice Score
Epoch
0.44
0.42
0.40
0.38
0.36
0.34
0.32
0
2
4
6
8


Pixel Accuracy vs Epoch
train
val
Pixel Accuracy
Epoch
0.70
0.69
0.68
0.67
0.66
0.65
0
2
4
6
8

Challenges & Solutions

- **Dataset Directory Mismatch**
  - *Challenge:* Training and testing scripts required a strict folder structure, causing errors when paths did not match.
  - *Solution:* Dataset folders were reorganized to exactly match the expected structure (Color Images and Segmentation).

- **Missing Ground Truth Masks**
  - *Challenge:* Some test images lacked corresponding segmentation masks, leading to evaluation crashes.
  - *Solution:* Dataset consistency was verified and missing references were handled to allow smooth evaluation.

- **CPU-Only Training and Inference**
  - *Challenge:* Lack of GPU significantly increased training and evaluation time.
  - *Solution:* Batch-wise processing and progress tracking were used to manage long execution times efficiently.

- **xFormers Dependency Warnings**
  - *Challenge:* Warnings appeared while loading the DINOv2 backbone due to unavailable xFormers support.
  - *Solution:* Verified that warnings were performance-related and did not affect model correctness.

- **Moderate IoU Performance**
  - *Challenge:* Achieved IoU (~0.31) was limited due to complex off-road terrain and ambiguous boundaries.
  - *Solution:* Emphasized Dice score and qualitative visual analysis to better assess segmentation quality.

- **Class Imbalance in Dataset**
  - *Challenge:* Dominance of background pixels biased predictions.
  - *Solution:* Dice metric was used alongside IoU to handle imbalance and evaluate segmentation performance fairly.

- **Output File Management**
  - *Challenge:* Large number of generated prediction files made organization difficult.
  - *Solution:* Outputs were structured into separate folders for binary masks, colored masks, and comparison images.

Conclusion & Future Work

Conclusion

In this project, an off-road semantic segmentation model was successfully developed using a DINOv2 vision transformer backbone with a custom segmentation head.

The model was trained on a multi-class off-road dataset containing terrain types such as trees, bushes, rocks, sky, and ground clutter.

The training process showed consistent improvement across epochs, with decreasing loss and increasing IoU, Dice score, and pixel accuracy.

Final evaluation achieved:

- **Validation Dice Score:** ~0.44
- **Pixel Accuracy:** ~70%
- **Best Validation IoU** ~0.30–0.31
- **Validation Accuracy** ~ 71%
- **Epoch reached** ~18

Qualitative results (colored mask predictions) demonstrate that the model can correctly identify major terrain regions, though fine boundary segmentation remains challenging.

Overall, the project demonstrates the effectiveness of self-supervised pretrained models for semantic segmentation with limited task-specific training.

Future Scope

- Train the model on a GPU with more epochs to significantly improve accuracy and IoU.

- Apply data augmentation techniques (random crops, flips, color jitter) to improve generalization.
- Experiment with advanced loss functions (Dice Loss, Focal Loss) to handle class imbalance.
- Fine-tune larger DINOv2 variants (Base/Large) for better feature representation.
- Extend the model for real-time inference in autonomous navigation or robotics applications.