TechnoMania 2.0: The Hackathon

x

Duality AI's

Offroad Semantic Scene Segmentation & Integrated Autonomy

1. Overview

Duality AI is excited to present the **Integrated Autonomy Challenge**. This competition is designed to explore cutting-edge AI training techniques by moving beyond simple "vision" into a specialized autonomous pipeline. Participants will train a model using annotated images of a desert environment, then test that model on a novel, but still desert, environment. All the data is generated from Duality AI's digital twin simulation platform, Falcon, using FalconCloud's geospatial-based digital twin environments. Each team will work to achieve the most accurate and precise model by adjusting training parameters and processes.

Along the way, participants will learn how Duality AI uses industry-proven techniques and tools and high-quality synthetic data to train AI models for context shifts, unseen environments, or difficult-to-access data, such as remote areas. Through this competition, participants will develop novel AI training skills, enhance their portfolio, build connections with Duality AI and other participants, and have the opportunity to win prizes and recognition

The Workflow:
All teams start with a common Foundation. Once the vision foundation is established, teams will **choose one of four specialized elective tracks** to implement: **Natural Language Processing (NLP)**, **Advanced Computer Vision (CV)**, **Generative AI (Gen AI)**, or **Speech-to-Text (STT)**.

Objectives

- **Foundation (Required):**
    - Train a robust semantic segmentation model using the provided synthetic dataset to accurately segment an environment, which is crucial to off-road autonomy
    - Evaluate model performance in novel (but similar) scenarios.
    - Benchmark and optimize your model to improve accuracy, generalizability, and

efficiency for real-world deployment scenarios.

- **Specialized Problem Statement (Choose One):**
  - **NLP:** Generate human-readable scene summaries and reasoning based on visual data.
  - **CV:** Implement autonomous path planning and obstacle avoidance using segmentation masks.
  - **Gen AI:** Use generative models to synthesize "Edge Case" scenarios and improve model robustness.
  - **Speech-to-Text:** Enable voice-activated commands to interact with and filter the UGV's visual perception.

Importance of Digital Twins for Segmentation and Off-road Autonomy Unmanned

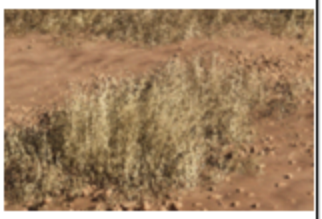
Ground Vehicles UGVs rely on robust computer vision models for effective path planning and decision-making in complex, dynamic environments. Among critical CV tasks, semantic segmentation plays a pivotal role in obstacle avoidance by providing fine-grained scene understanding. However, training high-performing segmentation models traditionally relies on supervised learning, which requires large, labeled datasets for effective training. Attaining this data in the real world, with necessary volume and variety for successful training, is generally costly and time-consuming, and often still proves insufficient.

Synthetic data presents a promising solution, and has already been explored as a means to address data scarcity and improve model robustness in computer vision tasks. Creating the data costs much less and takes very little time compared to traditional data collection and annotation methods. Additionally, users can control variations and edge cases such as weather events, time of day, and specific environment characteristics, providing diverse data for robust training.

2. Data Overview

Participants will work with a dataset generated from FalconEditor representing various desert environments.

Classes:

ID	Class Name	Picture
100	Trees	
200	Lush Bushes	
300	Dry Grass	
500	Dry Bushes	
550	Ground Clutter	

600	Flowers	
700	Logs	
800	Rocks	
7100	Landscape	*all general ground that isn't another category*
10000	Sky	

Participants will process synthetic data, train an AI model to segment the data images, validate performance on unseen images from a separate desert environment, and optimize accuracy under realistic constraints.

3. The Hackathon Structure

To maximize efficiency, we recommend dividing responsibilities based on these roles. Proper delegation will help streamline the workflow and ensure high-quality results across the technical and presentation components.

A. AI Engineering

Phase 1: The Foundation (Mandatory for All)

- Train and fine-tune the model using the generated dataset.
- Evaluate model performance.
- Use optimization techniques to improve accuracy and reduce inference time.

Phase 2: The Elective Track (Choose ONE)

Teams must select **one** of the following four tracks to build on top of their segmentation model:

Option A: NLP (Scene Reasoning)

- **Task:** Build a module that takes segmentation metadata (class counts/densities) and generates a text-based "Situational Report."
- **Example:** If "Rocks" occupy >15% of the frame, the model reports: *"High hazard density detected; proceeding with caution."*
- **Deliverable:** nlp_reasoner.py script.

Option B: Computer Vision (Path Planning)

- **Task:** Use the "Landscape" (7100) masks to calculate the safest drivable trajectory through the terrain.
- **Example:** Identify the largest contiguous area of "Landscape" and project a 2D "Path Vector" that avoids "Rocks."
- **Deliverable:** path_planner.py script with visual overlays.

Option C: Generative AI (Domain Robustness)

- **Task:** Identify images where your segmentation model fails (low IoU) and use GenAI to synthesize training data to fix the gap.
- **Example:** Use a Diffusion or GAN model to create "Sandstorm" or "Sunset" versions of

the training images to retrain the foundation.
- **Deliverable:** gen_augmenter.py and an "Improvement Report."

Option D: Speech-to-Text (Voice UI)

- **Task:** Implement a voice-command interface to filter the UGV's visual perception in real-time.
- **Example:** Operator says *"Show me the logs,"* and the UI highlights only Class 700 while dimming other segments.
- **Deliverable:** voice_ui.py script utilizing a speech recognition engine.

B. Documentation & Presenting Final Team Results

- Document the workflow, including:
  - Data augmentation/filtering strategies
  - Model training
  - Evaluation metrics such as loss graphs
- Prepare a structured report and final presentation showcasing findings, results, and insights.
- Create visualizations such as performance and loss graphs to highlight model behavior.

4. Key Deliverables

At the end of the hackathon, teams must submit:

1. **A Trained Semantic Segmentation Model:**
    a. Fully trained model that segments the testing images into categories.
    b. The package must include model weights, scripts, and config files.

2. **Specialized Tool:**

    The script associated with your chosen Elective Track.

3. **Performance Evaluation & Analysis Report**

    including but is not limited to:
    a. IoU Score to evaluate model accuracy.
    b. Loss graphs to visualize performance.
    c. Failure Case Analysis, highlighting misclassifications and possible improvements.

4. **Important links**

USE	LINK
Create a free Falcon account	Create Account - FalconCloud by Duality AI
Download the dataset	Hackathon Dataset
Submission form	Submit your results
Discord Forum	Join the Duality Discord

5. Judging Criteria (150 Point Scale)

- **Foundational Model Performance (IoU):** 80 Points
  - Measures the accuracy of the base segmentation model.
- **Elective Track Innovation:** 50 Points
  - 20 points for Innovation | 20 points for Technical Implementation | 10 points for Presentation
  - How effectively the chosen branch builds upon the segmentation output.
  - Technical complexity and utility of the elective feature.
- **Structured Findings & Detailed Reporting:** 20 Points
  - Well-organized documentation of the methodology, challenges, and solutions.
  - Clearly outlined steps, including any dataset modifications, model training, and evaluation.

Teams must balance technical performance with clear and professional documentation to maximize their score.

6. Task Instructions

This section will walk you through:

➔ Setting up your Falcon account
➔ Downloading and using the dataset
➔ Preparing your training environment
➔ Understanding the training workflow

1. Create a Falcon Account:

a. Visit Falcon and sign up for an account if you don't have one
b. Once registered, log in to access datasets, exercises, and tools.

2. Download the Dataset

a. Download the dataset to your local machine (see the "Important Links" section). You will need to create a FREE Falcon account, if you haven't already.
   i. This page has all the resources for all the Duality Hackathon tracks. Make sure you navigate to the "Segmentation Track" section.
   ii. The dataset includes:
      1. Pre-collected rgb color and segmented images a. Separated into Train and Val folders
      2. RGB color images in the testImages folder, to evaluate how well your model performs on unseen images
      3. Sample train and test scripts
      4. Environment setup scripts
      5. A script to visualize the segmentation using high-contrast colors

3. Set Up the Training Environment

a. Make sure you have Miniconda or Anaconda installed, and open an Anaconda Prompt window
b. Navigate to the ENV_SETUP sub folder
c. Run the setup_env.bat file in the Anaconda Prompt window
   i. This will set up an environment called "EDU", containing all the dependencies required to run the training and test scripts.

NOTE- Mac/Linux users, Create a setup_env.sh script with equivalent commands

4. Understand the Training Workflow

For these synthetic data competitions, the workflow varies from the traditional workflows:

a. train.py will train your model using the train and val images. The results of this step are important, but you will not know how robust your model is until you test it using unseen images.
b. test.py will test the model against images it HASN'T seen in training. The test images will be the same biome, but the actual location will be different.

5. Train the Model (of your choice) on the Sample Dataset

Once the EDU environment is ready and the dataset is in place:

a. Open an anaconda command prompt or a terminal
b. Navigate to the training and test scripts directory
c. Activate the environment by typing 'conda activate EDU' in the terminal.
d. Run the training command: 'python train.py' This will begin training your model and save logs + checkpoints to the runs/ directory.

6. Establish Benchmark Results on the sample dataset

After training is completed, evaluate your model's performance by running the train script (train.py) in the same command prompt window. This tests its performance on real-world test images and gives you the following:

a. Predictions
b. Loss metrics
c. IoU score

Use the results as your benchmark, so that later, when you train with newer model settings, you can:

- Compare performance
- Track improvements
- Identify where the model struggled (e.g., specific classes, specific locations)

Implement Elective:
Pass the segmentation masks or metadata into your specialized script (nlp_reasoner.py, path_planner.py, etc.).

7. Documentation & Presentation:

Ensure that your team's work is:

- ✔ Clear and understandable
- ✔ Reproducible by others
- ✔ Professional and impactful for judges

1. Keep it Structured & Organized using this General Structure

a. Title & Summary: Clearly state the purpose of the document.
b. Step-by-Step Instructions: Use numbered lists or bullet points.
c. Diagrams & Visuals: Use flowcharts, tables, and screenshots
d. Graphs & Charts: Show training loss, accuracy trends, and comparisons.
    i. Screenshots: Use images from the runs/ folder after training
    ii. Before & After Images: Show examples of correct vs. misclassified objects.

2. Document Everything, But Keep it Concise

a. Record major steps like dataset manipulation, training process, and evaluation.
b. Avoid overly technical language — aim for clarity.
c. Use clear, plain language—assume the reader is new to the project.

3. Example Entry:

a. Task: Model Training on Dataset
b. Initial IoU Score: 0.31
c. Issue Faced: Low recall for "Logs" class due to occlusion.
d. Solution: Augmented dataset with occlusion examples → New, better score

4. Documenting Failure Cases and Solutions

a. Include failure case images to illustrate what went wrong and how it was fixed.

5. Report Format (8 Pages Max)

Your Report should be concise, structured, and visually engaging. Use the following storytelling approach:

Problem → Fix→ Results → Challenges → Future Work

Page.No	Section	Description
1	Title	Team name, project name, brief tagline.
2	Methodology	Steps taken while training the model, and fine-tuned results.
3-4	Results & Performance Metrics	IoU score, confusion matrix, accuracy comparisons.
5-6	Challenges & Solutions	Key obstacles and how they were resolved.
7	Conclusion & Future Work	Final thoughts, and potential improvements.

8. Submission and Final Steps Instructions

i. Necessary Submission Files:

1. A single, Final Packaged Folder that includes all necessary files:
    a. Model training and inference scripts (train.py, test.py)
    b. configuration files
    c. Any additional assets or scripts required to test your model
2. A well-structured Hackathon Report PDF or DOCX that covers the following:
    a. Methodology: Your training approach and setup
    b. Challenges & Solutions: Issues faced and how you overcame them
    c. Optimizations: Techniques used to improve model performance
    d. Performance Evaluation: IoU score and Failure case analysis and observations.
3. A README.md or README.txt that provides:
    a. Step-by-step instructions to run and test your model
    b. How to reproduce your final results
    c. Any environment or dependency requirements
    d. Notes on expected outputs and how to interpret them

Note: You are welcome to use your own models and custom training scripts or notebooks. However, you must train your model exclusively on the dataset provided for this challenge.

Important: Using any of the designated testing images for training purposes is strictly prohibited and will result in disqualification. Please ensure a clear separation between training, validation, and test sets throughout your workflow.

ii. Upload Instructions:

1. Ensure your submission folder contains all of the above
2. Compress everything into a .zip file.
3. Upload the zipped folder to a private GitHub repository of your own.
4. Complete the submission form (see the “Important Links” section) :
    a. Reporting your team’s final score
    b. Providing the GitHub repository link
5. Finally, add the following reviewers as collaborators:
    a. Syed Muhammad Maaz
        i. GitHub Username - Maazsyedm
    b. Rebekah Bogdanoff
        i. GitHub Username - rebekah-bogdanoff
    c. Evan Goldman

i. GitHub Username - egol d010

iii. After Submission:

1. Sharing Results & Feedback
    a. After submission, teams can showcase their models and insights.
    b. Feedback from judges will be provided after evaluation.
2. Future Opportunities & Improvements
    a. Continue exploring advanced topics such as:
        i. Self-supervised learning
        ii. Domain adaptation
        iii. Multi-view detection
3. Stay connected with us via Discord for:
    a. Future challenges
    b. Internship and apprenticeship opportunities
    c. Community events and AI workshops

NOTE If you face any issues along the process join our discord channel linked above where we will be providing support to you for the hackathon.

9. Common Issues and Troubleshooting

FAQ's

1. Will setup_env.bat Work on Both Windows & Mac?
    a. No, The setup_env.bat script is designed primarily for Windows environments.
    b. Alternatively, for Mac/Linux use create a setup_env.sh shell script equivalent that installs all required packages.
2. My training process is too slow. What can I do?
    a. Here are a few tips to improve training speed:
        i. Reduce the batch size in your training configuration.
        ii. Close any unused applications or background processes to free up system resources.
        iii. If using GPU, monitor GPU usage with tools like nvidia-smi.
3. 3. How should I manage data transfer between team members or for submission?
    a. Why backup your data:
        i. Share results between teammates
        ii. Backup project files and checkpoints
    b. We recommend using cloud storage platforms such as Google Drive, Dropbox, OneDrive, git

10. Support and Communication

Join the official Duality Community Discord Server for:

- Real-time help
- Announcements
- Live Q&A with organizers
- Invite Link Here

We appreciate your hard work and look forward to reviewing your project! Feel free to share your work on LinkedIn and show the world your real-world problem solving skills in action!

Tag DualityAI to:

- Celebrate your team's efforts and creativity.
- Get noticed by industry experts and recruiters

— The Duality AI Team