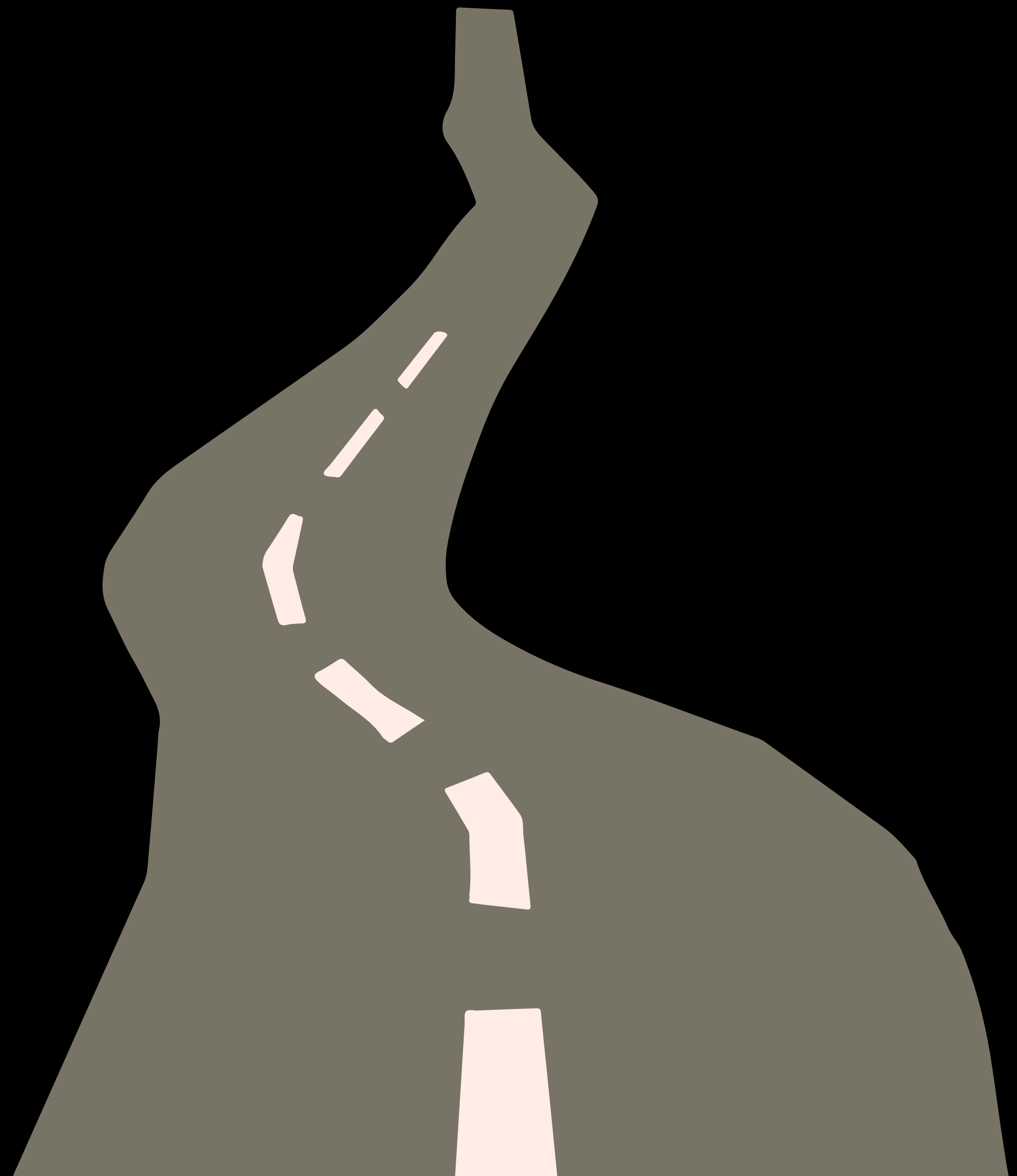
ROAD DAMAGE DETECTION ON ASU CAMPUS

Cohort 14444
CIS 515 : AI AI and Data Analytics Strategy

TEAM 103

INTRODUCTION



Problem

Undetected road cracks, potholes, and bumps pose a daily risk to students using bikes, scooters, and skateboards.

Why this?

Real incidents on campus have caused fractures and injuries. Most reports are manual and reactive.

Research Done

- Visual inspection of ASU campus routes
- Studied GitHub, Kaggle, Code Ninja and Papers with code datasets
- Reviewed Tempe 311 & ASU Fix-it portals

Goal

Build an automated CV-based system to detect damages and classify severity, assisting Facilities teams.

DON'T FALL FOR THE ROADS LIKE THEM

CURRENT APPROACH V/S OUR SOLUTION

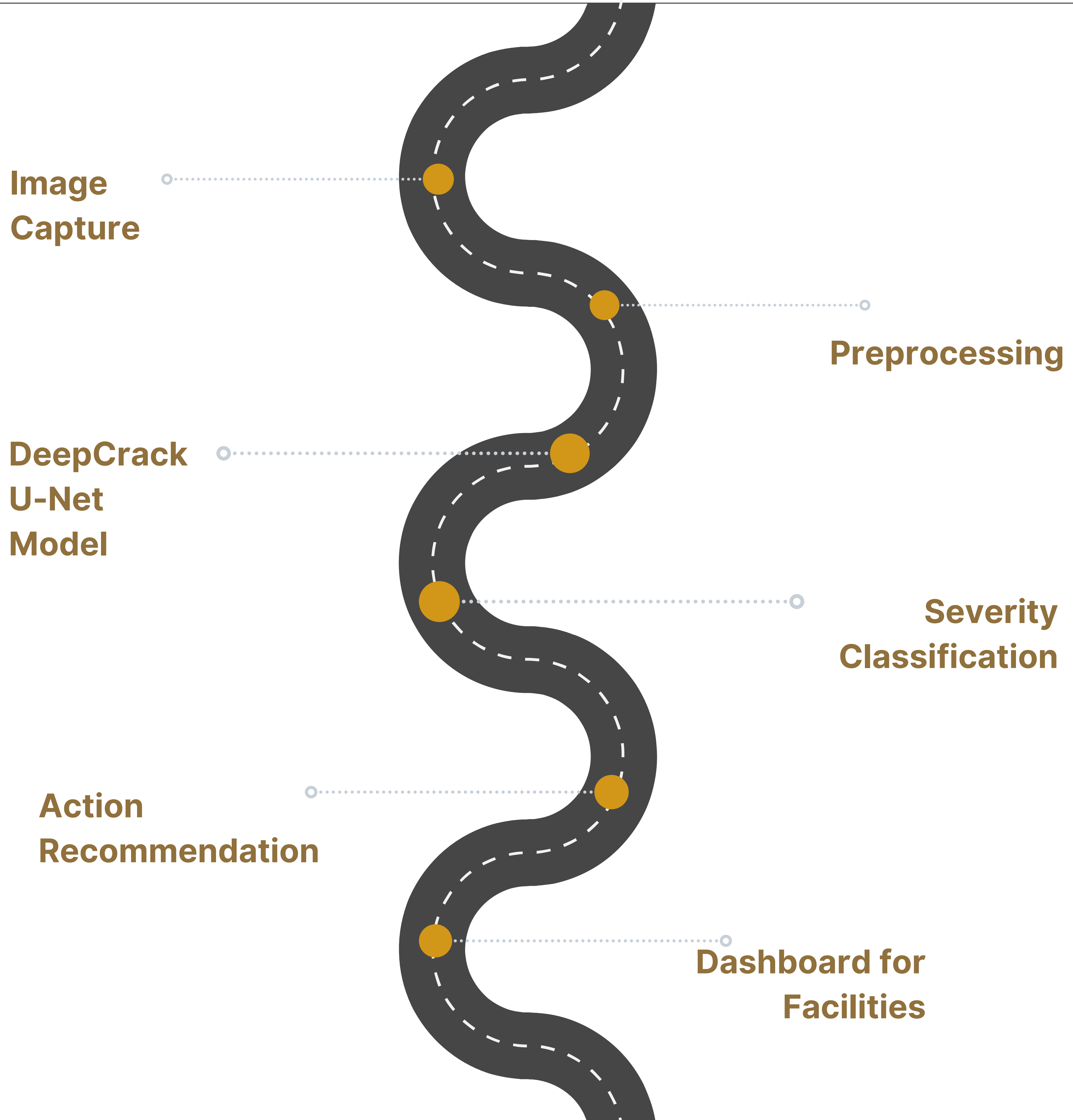
- Manual reporting by students and staff, Inspections
- Delayed repair, subjective severity judgment

Issues

- Inconsistent reporting
- Minor hazards/cracks often missed and ignored
- No severity-based action prioritization

Gaps - No automation, prioritization, or real-time alerts

Scope - Surface cracks only (not structural damage), ASU Tempe campus paths.



END-TO-END LIFECYCLE

- Collect images from ASU campus bike lanes, sidewalks
- Resize images to 255×255 pixels, normalize pixel values.
- Convert masks into binary labels (Crack vs No Crack)
- Use a customized U-Net architecture with VGG16 encoder
- Analyze the percentage of cracked area
- Generate action labels (Repair / Monitor / No Repair)
- Display predicted severity classifications in a dashboard

Crack % Thresholds

> 3% area: Immediate Repair
2–3%: Monitor
< 2%: No Repair

VALIDATION

Manual review of model predictions pre- and post-deployment

MONITORING

Retrain quarterly with new ASU data

UNINTENDED INCENTIVES

Risk: Over-prioritizing non-critical crack

Mitigation: Threshold tuning

PRIVACY

Only road surface images captured.

No identifiable personal data stored or processed.

RELATED PROJECTS

CRDDC2022 road damage challenge.

ArcGIS Road Surface Inspection

OUR DIFFERENCE

ASU-specific fine-tuning.

Severity classification customized for campus operations

DATASET AND MODEL TRAINING (80% Train, 20% Validation)

Dataset Sources

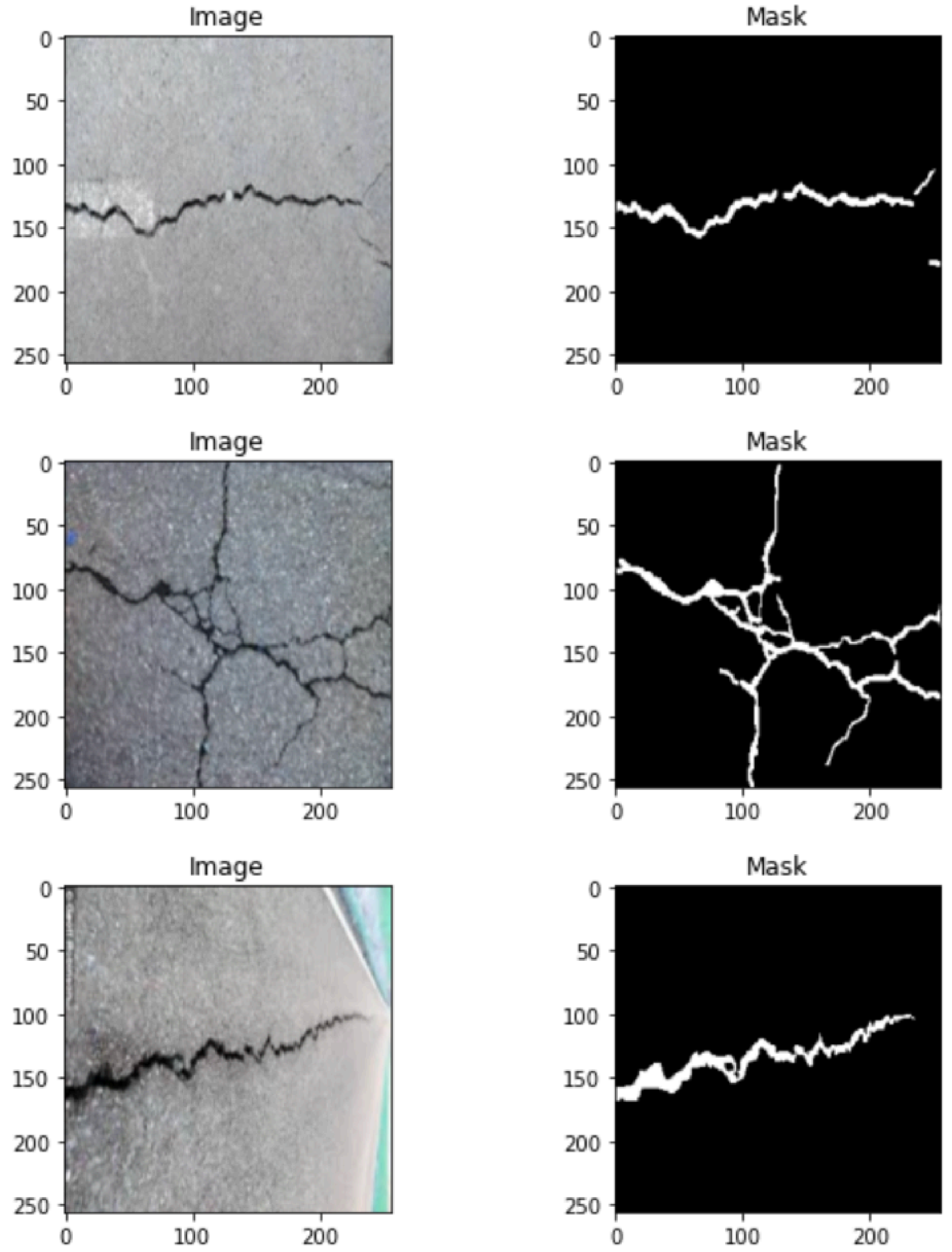
- 30+ ASU campus images (custom collected and labeled)
- DeepCrack open dataset (road cracks)

Preprocessing

- Images resized to **255×255**
- Masks converted to binary crack/no-crack labels
- Random horizontal and vertical flips for augmentation

Workflow Changes

- Weekly dashboard reviews.
- Crack severity used for ticket prioritization.



CV Model Architecture ?

CV Component	Damage detection and severity scoring
Why CV ?	Manual inspections unreliable
	Sensors cannot visually differentiate crack severity
Model	DeepCrack U-Net with VGG16 Encoder
Encoder	VGG16 pretrained on ImageNet (frozen weights)
Decoder	Upsampling + convolution + batch normalization
Output	Binary crack segmentation mask using sigmoid activation
Loss	Dice Loss
Metrics	Accuracy, Dice Coefficient
Size	Epochs: 15 and Batch Size: 8

Outcome	Action
True Positive	Immediate repair scheduling
True Negative	No action needed
False Positive	Manual verification by Facilities
False Negative	Emergency manual reporting if injury occurs

Cracked_Pixels	Cracked_Pixels	Cracked_Pixels	Action_Recommended
IMG_0197.JPG	2822	4.31	⚠️ Immediate Repair Needed
IMG_0198.JPG	947	1.45	✅ No Repair Needed
IMG_0201.JPG	1437	2.19	🛠️ Monitor - Not Urgent
IMG_0202.JPG	736	1.12	✅ No Repair Needed

CSV output

DeepCrack Detection App

Upload a road image

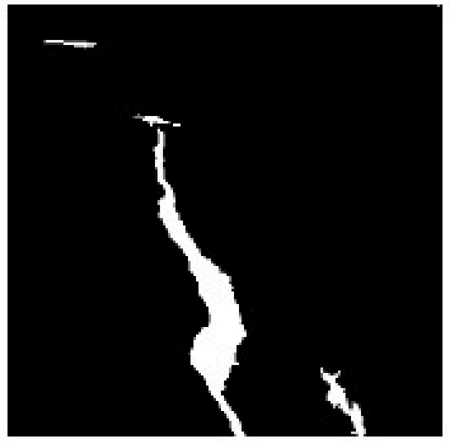
Drag and drop file here
Limit 200MB per file • JPG, PNG, JPEG

Browse files

WhatsApp Image 2025-04-25, 23.08.01_d07ec368.jpg 0.7MB

Original Image

Predicted Crack Mask



Overlayed Result

Crack Coverage: 4.17%

🚨 ⚠️ Immediate Repair Needed

DEMO

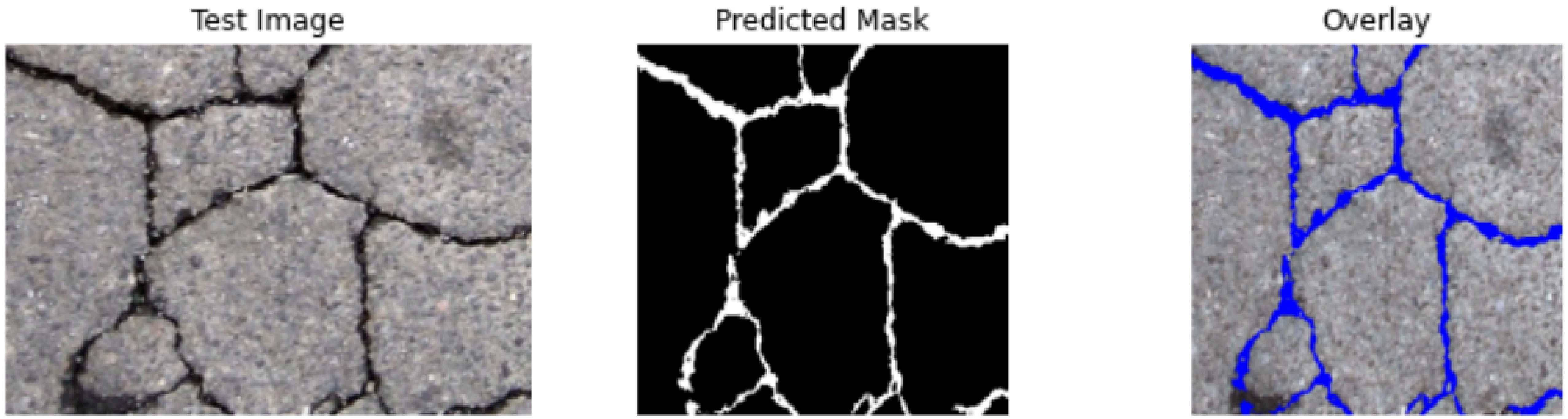
Success Metrics

~91% Validation Accuracy

~0.81 Validation Dice Coefficient

Severity match ≥ 85% (vs manual validation)

Proof of Concept :
Input Image → Predicted Mask → Overlay Visual



risk & bias

Daylight Bias

- Most ASU images collected in daytime; model less effective at night or under poor lighting.

Surface Texture Confusion

- Shadows, painted lines, and surface stains could be falsely detected as cracks.

Class Imbalance

- Dataset has more crack examples than potholes or speed bump defects.

Deployment Risk

- High false positives may overload Facilities team with unnecessary alerts.

mitigation strategy

Data Augmentation

- Apply brightness, contrast variations to simulate night and shaded conditions.

Manual Validation Layer

- Facility team manually reviews critical alerts for top-priority repairs.

Class Weighting During Training

- Balance impact of rare classes during loss calculation.

Periodic Retraining

- Collect more diverse ASU images across different times, seasons, and surfaces every 3 months.

Validation & Monitoring

Validation Pre-Deployment

- Compare model predictions on 30+ ASU images against manually created masks.
- Severity classification validated by a facilities management volunteer review

Validation Post-Deployment

- Facilities track the number of incident reports before and after CV deployment.
- Crack detection alerts cross-verified with field inspections monthly.

Model Monitoring Plan

- Log prediction confidence scores.
- Retrain model quarterly with new diverse lighting/weather conditions

Updating Mechanism

- Image data continuously expanded by campus operations team.
- Review model thresholds every 6 months based on field repair feedback.

Beneficiaries

ASU students, visitors, campus operations

Value Delivered

Safer mobility, Faster maintenance,
2–3 accidents avoided monthly

Stakeholders

ASU Facilities, Health Services, ASU Police,
Tempe Public Works

Cost Benefit Estimate

~$1000/month savings in potential injury costs

We estimated 2–3 mobility-related accidents monthly based on ASU campus data and typical ER treatment costs (~$300–$600 per case), leading to an approximate ~$1000/month in potential savings.

LEARNT

Lessons

- Manual labeling is labor-intensive but critical.
- Labeling campus data is time-consuming
- Batch inference & Pretrained models (VGG16) speed up development and deployment.

Limitations

- Shadows and paint marks can trigger false positives.
- Limited night-time crack detection.

Mitigations

- Dataset augmentation with different lighting/weather conditions.

Scalability

- Expandable to downtown Phoenix campus, Polytechnic campus.
- Scalable to sidewalks, parking lots, and even neighboring Tempe areas.

INSIGHTS & RECOMMENDATIONS

Short Term:

- Deploy dashboard integration with ASU Facilities.

Mid Term:

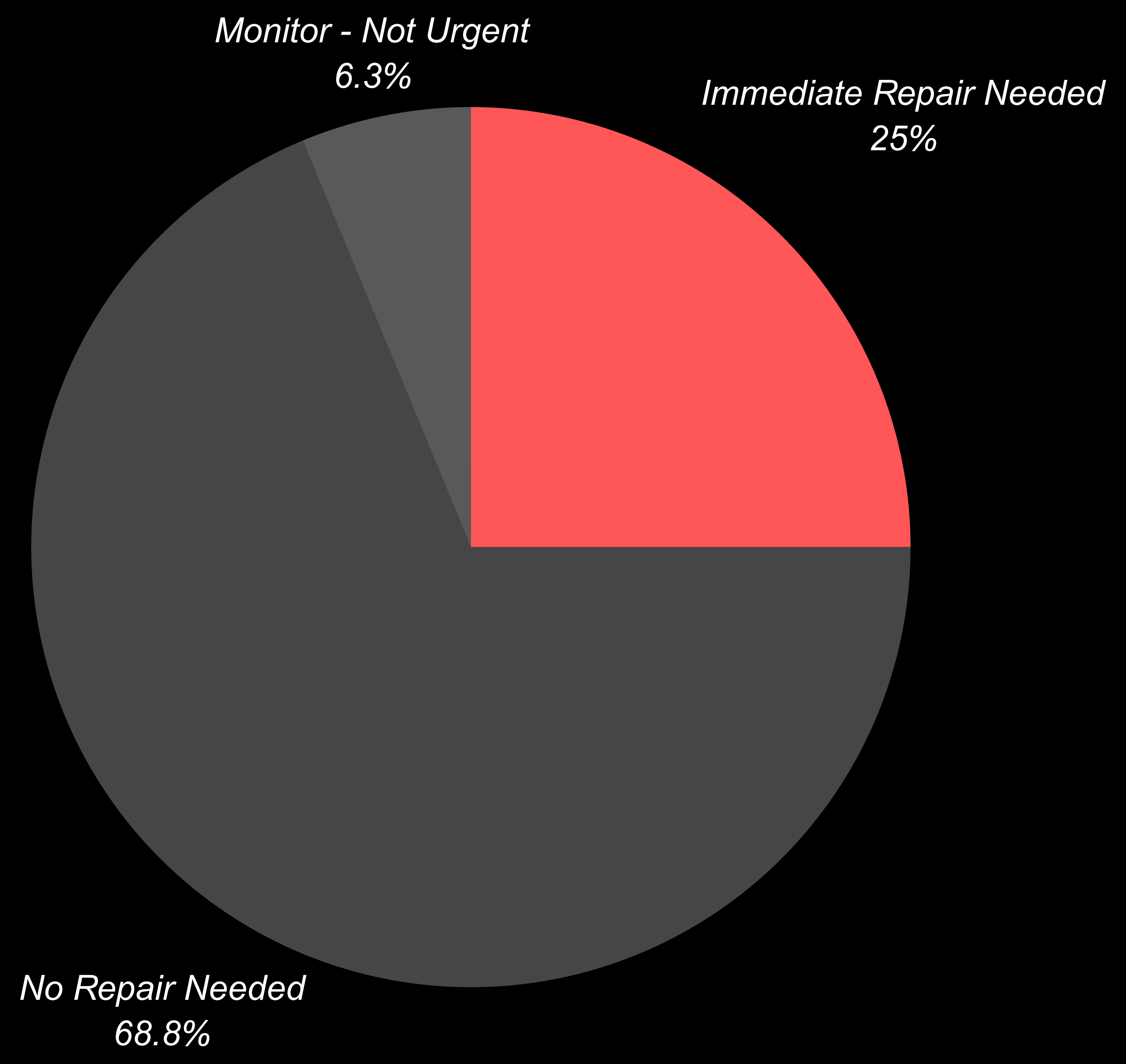
- Extend to sidewalks, ramps, parking lots.
- Location tracker when photos clicked.

Long Term:

- Develop a mobile app for real-time student reporting.
- Use drones for larger areas.

Model Enhancements:

- Add multi-class labeling (potholes, bumps, surface cracks)
- Handle night-time or rain conditions better.



Road Damage Severity Distribution

SUMMARY AND FUTURE WORK

Summary of Solution:

- Developed an automated CV-based pipeline to detect surface cracks and recommend actions on ASU campus bike lanes and sidewalks.
- End-to-end system: Image Capture → Preprocessing → DeepCrack U-Net Model → Severity Classification → Action Recommendation → Dashboard Alerts.

Key Findings:

- Achieved **~91% validation accuracy and ~0.81 Dice Coefficient**.
- Successfully classified damage severity into Immediate Repair, Monitor, or No Repair categories.
- Model outputs matched manual inspection with ~85% accuracy.

Limitations:

- Reduced performance under night-time or poor weather conditions.
- Some false positives from shadows and paint lines.

Possible Extensions:

- Extend detection to sidewalks, parking zones, and ramps across all ASU campuses.
- Improve multi-class damage detection (distinguish potholes, cracks, speed bumps).

Future Work:

- Develop a mobile app for real-time student reporting.
- Integrate detection outputs directly into ASU’s Fix-it maintenance system.
- Incorporate drone-based aerial image capture for faster data collection.

thank you

Team 103

References

DeepCrack Official GitHub Repository

DeepCrack: A deep learning-based crack detection framework.
GitHub Repository

Streamlit Framework for Web App DEMO Deployment

Streamlit Inc. "Streamlit: The fastest way to build and share data apps."