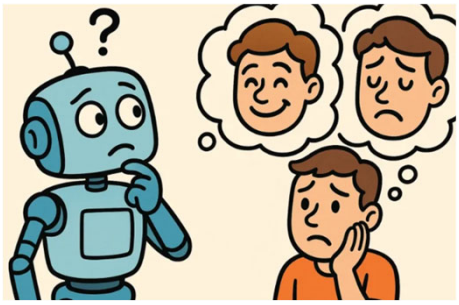
Soft computing

K.N. Toosi University of Technology

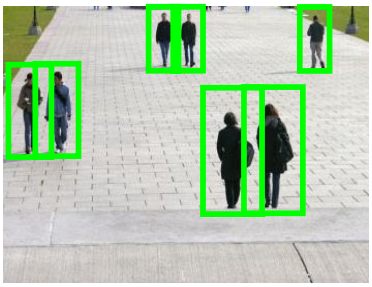
تعریف بینایی ماشین

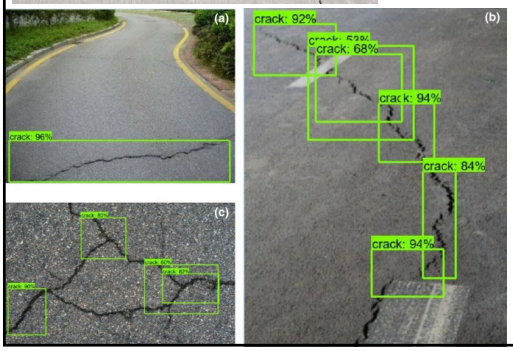
- بینایی ماشین شاخه‌ای از هوش مصنوعی است که به استخراج اطلاعات از تصاویر یا ویدیوها می‌پردازد.
- هدف :درک و تفسیر محتوای دیداری توسط رایانه‌ها، مشابه مغز انسان.



Soft Computing 3

کاربردهای اصلی بینایی ماشین



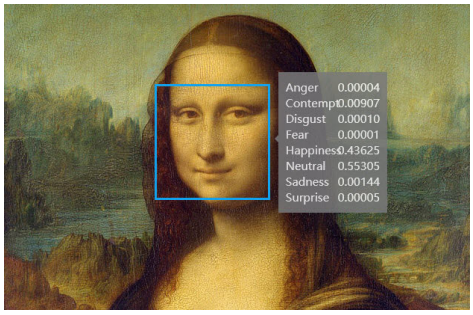


- تشخیص چهره
- پلاک‌خوانی خودروها
- تشخیص اشیا
- طبقه‌بندی تصاویر
- آنالیز ویدیو
- تشخیص عیوب در تولید صنعتی
- نقشه‌برداری و مدل‌سازی سه‌بعدی

Soft Computing 4

مراحل اصلی در بینایی ماشین

- پیش‌پردازش تصویر (Preprocessing)
- استخراج ویژگی‌ها (Feature Extraction)
- یادگیری و طبقه‌بندی (Learning & Classification)
- تحلیل و تصمیم‌گیری



https://www.projectoxford.ai/demo/Emotion#detection

روش‌های اصلی در بینایی ماشین

- روش‌های کلاسیک :فیلترها، تشخیص لبه، الگوریتم‌های مورفولوژیکی
- یادگیری ماشین: SVM ، KNN، درخت تصمیم
- یادگیری عمیق: CNN ، R-CNN، YOLO، U-Net

روش‌های کلاسیک- فیلترها

فیلترها

فیلتر یا Kernel یا Mask ماتریس کوچکی است که روی تصویر اصلی حرکت می‌کند (عمل کانولوشن) تا ویژگی‌هایی مثل لبه، بافت، تاری یا تیزی را استخراج کند.

مثال:
یک فیلتر ۳×۳ (مانند فیلتر تیزی) ممکن است به شکل زیر باشد

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

این فیلتر تصویر را **تیز (Sharpen)** می‌کند.

روش‌های کلاسیک- تشخیص لبه

تشخیص لبه

لبه‌ها مناطقی هستند که شدت پیکسل‌ها به‌صورت ناگهانی تغییر می‌کند (مثل مرز بین دو رنگ یا شیء). تشخیص لبه برای مشخص کردن شکل اشیاء، استخراج ویژگی‌ها برای طبقه‌بندی یا شناسایی و فشرده‌سازی اطلاعات تصویر

الگوریتم	ماتریس فیلتر	ویژگی
Sobel	محاسبه گرادیان در X و Y	تشخیص لبه‌های افقی و عمودی
Prewitt	مشابه Sobel ولی ساده‌تر	کمتر حساس به نویز
Laplacian	فیلتر دوم مشتق	تشخیص لبه در تمام جهت‌ها
Canny	چند مرحله‌ای، دقیق‌تر	کاهش نویز، تشخیص قوی لبه‌ها

مثال فیلتر Sobel X

$$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

مثال فیلتر Sobel Y

$$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

روش‌های کلاسیک- الگوریتم‌های مورفولوژیکی

الگوریتم‌های مورفولوژیکی

این الگوریتم‌ها بر روی تصاویر **باینری (سیاه و سفید)** استفاده می‌شوند، و ساختار و شکل اجسام را بررسی و اصلاح می‌کنند.

عملیات	کاربرد	شرح
فرسایش (Erosion)	حذف نویزهای کوچک	نقاط سفید نزدیک به لبه را حذف می‌کند
انبساط (Dilation)	پر کردن حفره‌ها	نواحی سفید گسترش می‌یابند
باز شدن Opening	حذف نویز و حفظ ساختار اصلی	Erosion → Dilation
بسته شدن Closing	پر کردن شکاف‌ها	Dilation → Erosion

کاربردها

- ✓ حذف نویز از تصویر مثلا ترک بتن
- ✓ پر کردن شکاف در تصاویر ترک برای شمارش طول یا سطح
- ✓ تعیین مرز ترک‌ها

شبکه عصبی مصنوعی کانولوشنی

CNN: Convolutional Neural Networks

نوعی از **شبکه‌های عصبی مصنوعی** است که می‌تواند داده‌های تصویری را **به‌صورت خودکار تحلیل و یاد بگیرد**. برخلاف شبکه‌های عصبی سنتی که نیاز به پیش‌پردازش دستی ویژگی‌ها دارند، CNNویژگی‌های مهم تصویر را به‌صورت خودکار استخراج می‌کند

مؤلفه	توضیح
لایه کانولوشن Convolution Layer	فیلترهایی (Kernel) را روی تصویر حرکت می‌دهد تا ویژگی‌هایی مثل لبه، گوشه یا بافت را شناسایی کند.
لایه فعال‌سازی ReLU	عملیات غیرخطی (مثل ReLU: max(0, x)) برای افزایش توانایی مدل در یادگیری روابط پیچیده.
لایه زیرنمونه‌گیری Pooling	اندازه تصویر را کاهش می‌دهد و به مدل کمک می‌کند تا نسبت به جابه‌جایی‌ها مقاوم شود.
لایه اتصال کامل Fully Connected	پس از استخراج ویژگی‌ها، تصمیم‌گیری نهایی (مثلاً طبقه‌بندی) را انجام می‌دهد.
Softmax	خروجی نهایی برای دسته‌بندی (مثلاً احتمال تعلق تصویر به هر کلاس).

شبکه عصبی مصنوعی کانولوشنی

R-CNN: Region-based Convolutional Neural Networks

یک مدل یادگیری عمیق برای تشخیص و تعیین محل اشیاء در تصویر است. توسط Ross Girshickدر سال ۲۰۱۴ معرفی شد.
برخلاف CNNکه فقط کلاس تصویر را تشخیص می‌دهد، R-CNN موقعیت (مختصات) اشیاء را هم تعیین می‌کند.
مثال: این تصویر شامل "خودرو" است و مختصات آن در تصویر برابر است با (x, y, width, height)

سه مرحله اصلی R-CNN

۱. پیشنهاد ناحیه (Region Proposal)
- تصویر ورودی را به حدود ۲۰۰۰ ناحیه پیشنهادی تقسیم می‌کند (با استفاده از الگوریتم (Selective Search
- این نواحی ممکن است شامل اشیاء مختلف باشند.

۲. ویژگی‌برداری با CNN:
- هر ناحیه پیشنهادی به شبکه CNNداده می‌شود تا ویژگی‌های آن استخراج شود.

۳. طبقه‌بندی و تعیین جعبه مرز (Bounding Box)
- از SVM برای تشخیص کلاس شیء در هر ناحیه استفاده می‌شود.
- از مدل رگرسیون خطی برای اصلاح مختصات ناحیه استفاده می‌شود.

شبکه عصبی مصنوعی کانولوشنی

R-CNN: Region-based Convolutional Neural Networks

نقاط ضعف R-CNN

- کند و پرهزینه: هر تصویر باید برای حدود ۲۰۰۰ ناحیه به CNNداده شود.
- غیر بهینه: هر مرحله (CNN, SVM, Regression)جداگانه آموزش می‌بینند.

نسخه‌های بهبود یافته	بهبود
Fast R-CNN	ویژگی‌ها را فقط یک بار از کل تصویر استخراج می‌کند
Faster R-CNN	شبکه‌ای برای تولید نواحی پیشنهادی می‌سازد (Region Proposal Network)
Mask R-CNN	علاوه بر تشخیص، نقشه‌بندی دقیق پیکسل‌ها(Segmentation) را نیز انجام می‌دهد

کاربردها

- ✓ شناسایی ترک‌ها یا حفره‌ها در تصاویر بتن یا آسفالت
- ✓ تشخیص وسایل نقلیه در مسیر پروژه‌های عمرانی
- ✓ شمارش تیرهای بتنی یا ستون‌ها در تصاویر هوایی

الگوریتم تشخیص اشیا

YOLO: You Only Look Once

الگوریتم YOLO برخلاف روش‌های سنتی که چندین بار روی تصویر نگاه می‌کنند، فقط یک‌بار تصویر را پردازش می‌کند،

YOLO تصویر را به یک شبکه مربعی $S \times S$ تقسیم می‌کند. هر سلول:
احتمال وجود شیء را پیش‌بینی می‌کند.
مختصات آن شیء را ارائه می‌دهد: (x,y,width,height)
همچنین احتمال تعلق آن به کلاس‌هایی مثل "ترک"، "ماشین"، "آدم" و غیره.

نسخه	ویژگی
YOLOv1	نسخه اولیه، ساده ولی سریع
YOLOv3	بسیار محبوب و دقیق‌تر
YOLOv4	بهبودهای بزرگ در دقت و سرعت
YOLOv5	سبک، انعطاف‌پذیر، پشتیبانی قوی در PyTorch
YOLOv8	آخرین نسخه توسط Ultralytics – پشتیبانی از تشخیص، سگمنتیشن و ...

کاربردها

- ✓ ترک‌های سطحی را تشخیص دهد.
- ✓ شدت و محل آن‌ها را روی تصویر مشخص کند.
- ✓ در سیستم‌های هوشمند نگهداری و پایش سازه‌ها (SHM) استفاده شود.

نصب کتابخانه

```
pip install ultralytics
```


مثال الگوریتم تشخیص اشیا

YOLO: You Only Look Once

الگوریتم YOLO برخلاف روش‌های سنتی که چندین بار روی تصویر نگاه می‌کنند، فقط یک‌بار تصویر را پردازش می‌کند،

```
from ultralytics import YOLO

# بارگذاری مدل
model = YOLO("yolov8n.pt")

# اجرا روی تصویر ترک
results = model("steel_crack.jpg", save=True)
```

YOLOv8n

با دیتاست COCO ارایه شده توسط مایکروسافت آموزش دیده است که هشتاد شی معمول انسان، ماشین، سگ، دوچرخه ... را تشخیص می‌دهد.
برای تشخیص ترک بایستی آموزش ببینند.

COCO Data set: https://cocodataset.org/#download

مثال

R-CNN: Region-based Convolutional Neural Networks

تشخیص اشیای تصویر با استفاده از Fast R-CNN .

نصب کتابخانه ها

```
pip install opencv-python
pip install torch torchvision torchaudio
```

نصب برنامه git و

https://git-scm.com/downloads/win

کامپایلر C++مثل Visual Studio Build Tools https://visualstudio.microsoft.com/visual-cpp-build-tools/

نصب کتابخانه

```
pip install git+https://github.com/facebookresearch/detectron2.git
```


مثال شبکه عصبی مصنوعی کانولوشنی

R-CNN: Region-based Convolutional Neural Networks

برنامه R-CNN.

```
import cv2
import torch
from detectron2.engine import DefaultPredictor
from detectron2.config import get_cfg
from detectron2 import model_zoo
from detectron2.utils.visualizer import Visualizer
from detectron2.data import MetadataCatalog

# تنظیمات مدل از پیش آموزش دیده‌شده Fast R-CNN (ResNet-50)
cfg = get_cfg()
cfg.merge_from_file(model_zoo.get_config_file("COCO-Detection/faster_rcnn_R_50_FPN_3x.yaml"))
cfg.MODEL.ROI_HEADS.SCORE_THRESH_TEST = 0.5  # آستانه اعتماد برای تشخیص‌ها
cfg.MODEL.WEIGHTS = model_zoo.get_checkpoint_url("COCO-Detection/faster_rcnn_R_50_FPN_3x.yaml")
cfg.MODEL.DEVICE = "cuda" if torch.cuda.is_available() else "cpu"
```

مثال شبکه عصبی مصنوعی کانولوشنی

R-CNN: Region-based Convolutional Neural Networks

برنامه R-CNN.

```
predictor = DefaultPredictor(cfg)

# 📷 خواندن تصویر ورق فولادی دارای ترک
image = cv2.imread("steel_crack.jpg")  # تصویر باید در مسیر پروژه باشد

# 🔍 تشخیص
outputs = predictor(image)

# 🖼️ نمایش نتایج با جعبه دور ترک‌ها
v = Visualizer(image[:, :, ::-1], MetadataCatalog.get(cfg.DATASETS.TRAIN[0]),
scale=1.2)
out = v.draw_instance_predictions(outputs["instances"].to("cpu"))

# نمایش تصویر نهایی
cv2.imshow("Detected Cracks", out.get_image()[:, :, ::-1])
cv2.waitKey(0)
cv2.destroyAllWindows()
```


مثال شبکه عصبی مصنوعی کانولوشنی

R-CNN: Region-based Convolutional Neural Networks

نتیجه برنامه R-CNN.

مثال آموزش U-net برای ترک

U-net یکی از مؤثرترین معماری‌ها به روش شبکه عصبی کانولوشنی برای تقسیم‌بندی تصاویر (Image Segmentation) است که مخصوصاً در حوزه‌هایی مانند پردازش تصاویر پزشکی، صنعتی (مثل شناسایی ترک) کاربرد دارد.

ساختار این مدل به‌صورت پایین‌رو – بالارو (encoder-decoder) کار می‌کند، و شکل Uدارد، به همین دلیل نام آن U-Net است.

۱ - **مسیر پایین‌رو (Encoder (downsampling**
هدف: استخراج ویژگی‌ها از تصویر.
شامل چندین بلوک کانولوشنی + ReLU + MaxPooling.
در هر مرحله اندازه تصویر کاهش و تعداد کانال‌ها افزایش می‌یابد.

۲ - **مسیر بالارو (Decoder (upsampling**
هدف: بازسازی تصویر با همان ابعاد اولیه اما همراه با نقشه‌ی بخش‌بندی (segmentation mask).
شامل لایه‌های upsampling (مثل ConvTranspose2dو کانولوشن.
در هر مرحله اندازه تصویر افزایش و تعداد کانال‌ها کاهش می‌یابد.

۳ - **اتصال‌های پرشی Skip Connections**
ویژگی کلیدی U-Net.
ویژگی‌های مسیر پایین‌رو مستقیماً به مرحله متناظر در بالارو داده می‌شوند.
این کار باعث حفظ جزئیات مکانی (spatial details) در خروجی می‌شود.

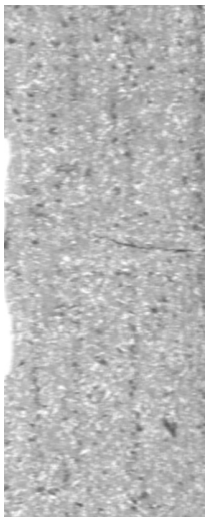
مثال آموزش U-net برای ترک

مزیت	توضیح
عملکرد عالی با دیتای کم	به‌خاطر skip connectionها و داده‌افزایی (data augmentation)
دقت بالا در مرزبندی‌ها	چون از اطلاعات سطح پایین تصویر استفاده می‌کند
ساده و قابل پیاده‌سازی	با PyTorch، TensorFlowو...
قابل تنظیم	تعداد لایه‌ها، فیلترها، سایز ورودی قابل تغییر است

برای تشخیص ترک این مدل با ۳۲۰ تصویر از دیتاست KolektorSDD آموزش و با ۸۰ تصویر تست شد که حدود یکساعت طول کشید

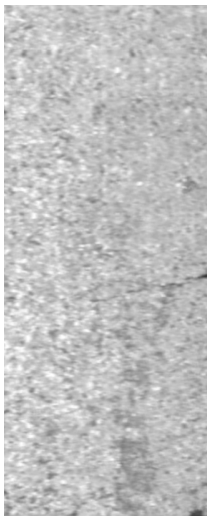
مثال آموزش YOLO, U-net برای ترک

نمونه تصاویر آموزش

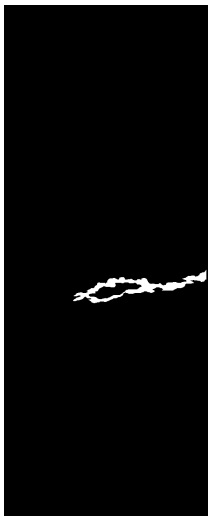


Part5.jpg

Mask:Part5_label.bmp

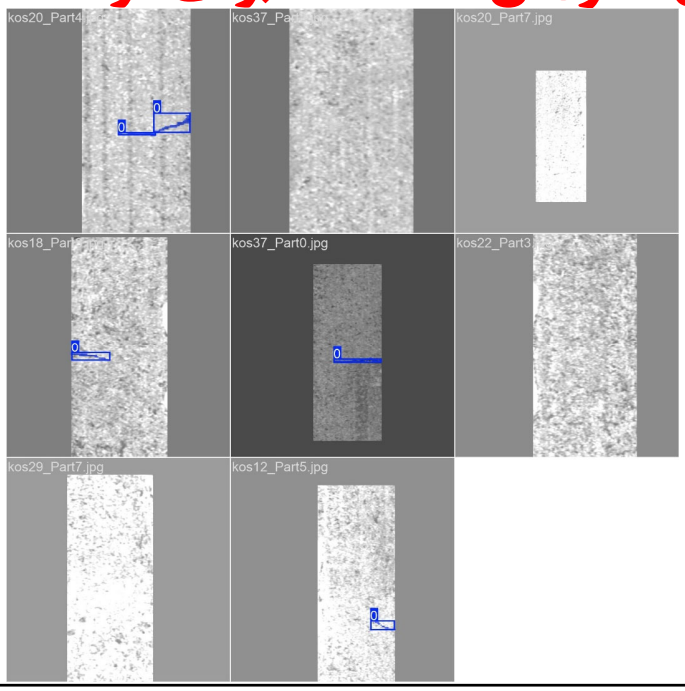


Part0.jpg



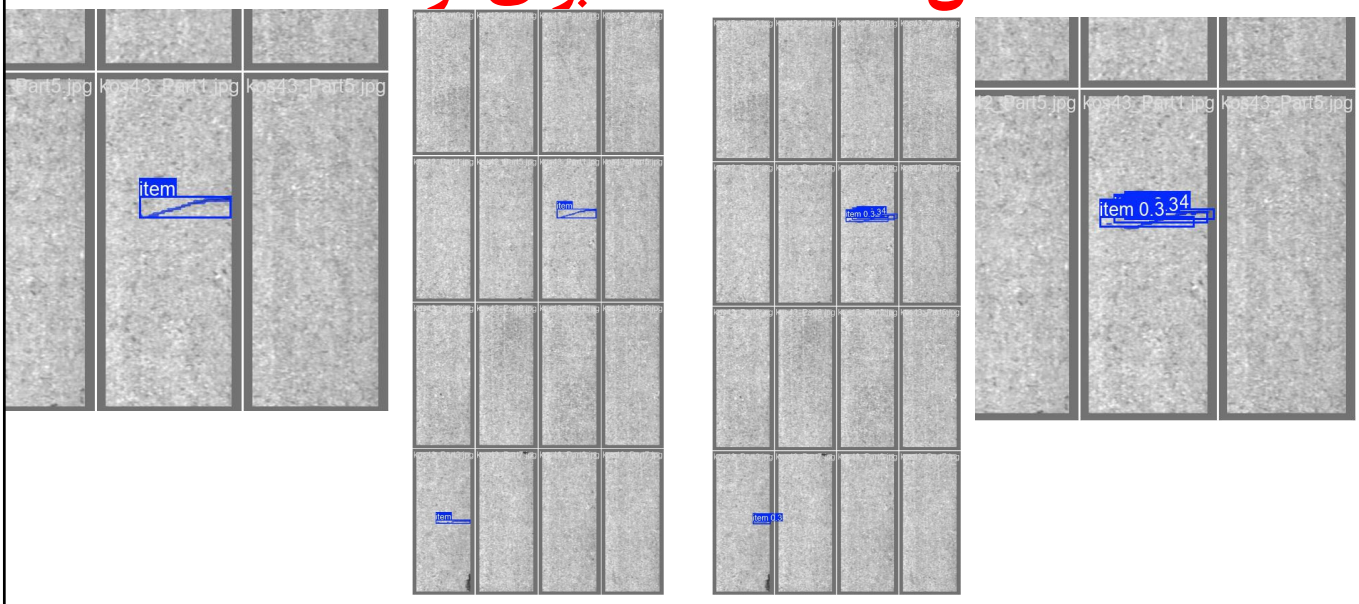
Mask:Part0_label.bmp

مثال آموزش YOLO برای ترک



نمونه نتایج آموزش

مثال تست YOLO برای ترک

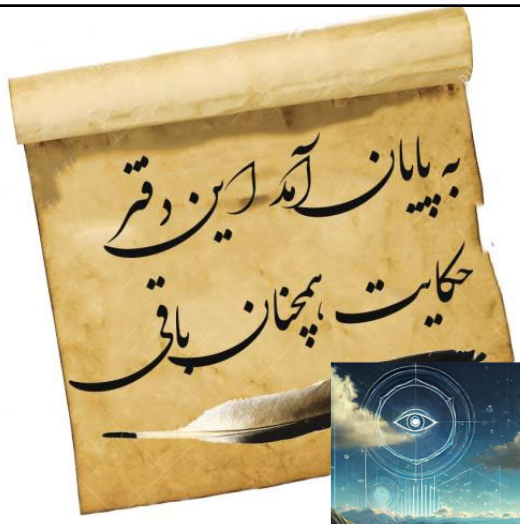


نمونه نتیجه تست نمونه برچسب ها

پروژه برنامه نویسی

تمرین : تعیین ترک در ورقهای فولادی یا صفحات بتنی با استفاده از بینایی ماشین

- جمع‌آوری داده‌ها: مجموعه‌ای از تصاویر صفحات بتنی یا فولادی دارای ترک و بدون ترک را از اینترنت یا پایگاه‌های داده (مثلاً Kaggle)جمع‌آوری کنید. تمامی داده ها را با از نظر ابعاد یکسان کنید.
- پیش‌پردازش تصویر : خواندن تصاویر با .OpenCVتبدیل به طیف خاکستری، افزایش کنتراست یا استفاده از فیلترهای آستانه‌ای (Thresholding) برای برجسته‌سازی ترک‌ها.اعمال فیلترهایی مانند Sobelیا Cannyبرای تشخیص لبه.
- استخراج ویژگی : می‌توانید از روش‌های ساده آماری یا متدهای پیشرفته‌تر مثل HOG، LBPیا CNNبهره بگیرید.
- مدل‌سازی : ساخت مدل طبقه‌بندی(Binary Classification) برای پیش‌بینی "ترک دارد" یا "ترک ندارد" با یکی از الگوریتم‌های زیر:Logistic RegressionRandom Forest(SVM) Support Vector Machineیا استفاده از یک شبکه عصبی کانولوشنی (CNN) ساده با Kerasیا PyTorch
- آزمون و ارزیابی: استفاده از معیارهایی مانند Accuracy، Precision، Recallو F1-scoreبرای ارزیابی مدل.نمایش نتایج به صورت تصویری با matplotlib.
- تشخیص ناحیه ترک : استفاده از مدل‌هایی مثل YOLOیا Detectron2برای تعیین موقعیت دقیق ترک در تصویر.
- **خروجی مورد انتظار**
  - فایل Pythonشامل کل مراحل پردازش تصویر و طبقه‌بندی
  - .نمودارهای ارزیابی عملکرد مدل
  - یک فایل گزارش PDFکوتاه شامل:مراحل کارنوع مدل استفاده‌شدهدقت مدل و تحلیل نتایج



به دنبال آرزوهایم خواهم رفت
مگر نمیرم عهد بسته ام قبل از

Soft Computing 25