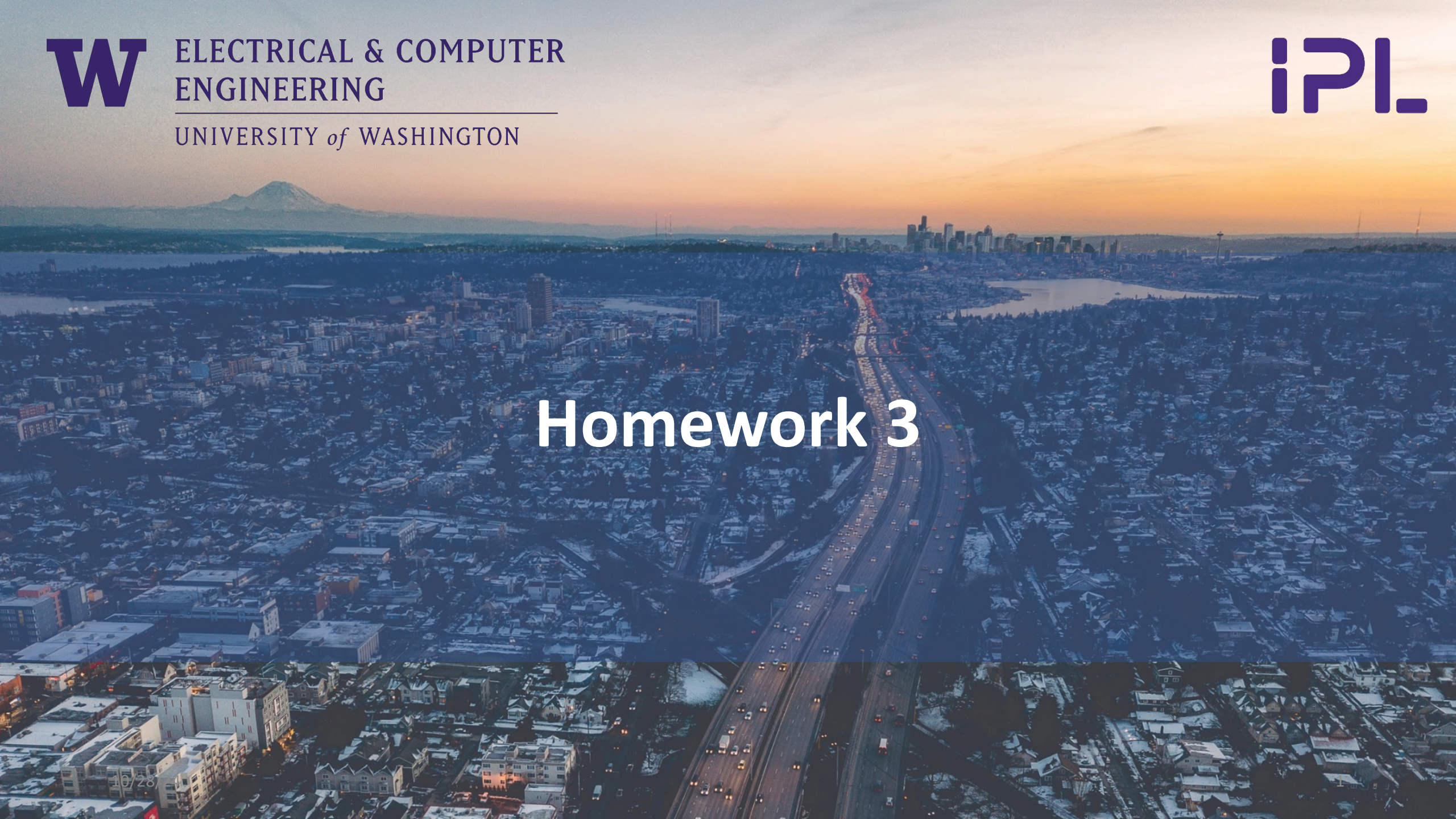
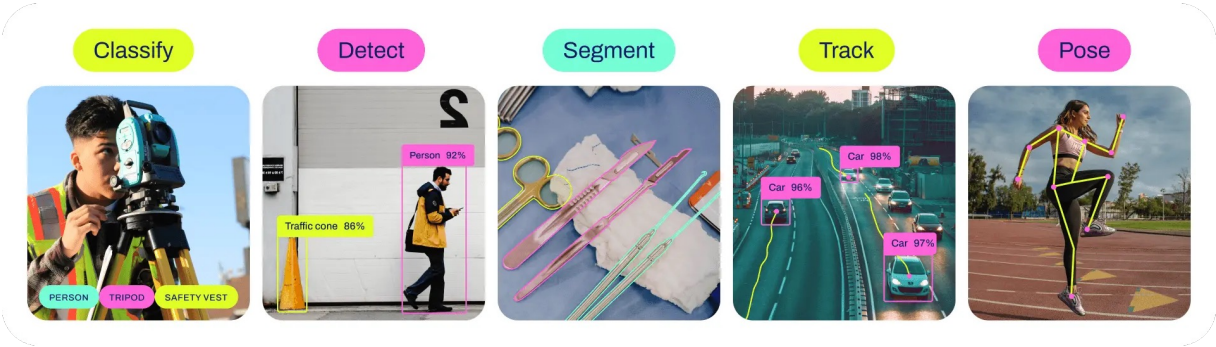

ELECTRICAL & COMPUTER ENGINEERING
UNIVERSITY of WASHINGTON
IPL
Homework 3
10/28

Preparation

- Please download everything from here: https://drive.google.com/drive/folders/19yCCwcE4j9wx6kipGGUUs-vnhDOmeELZ

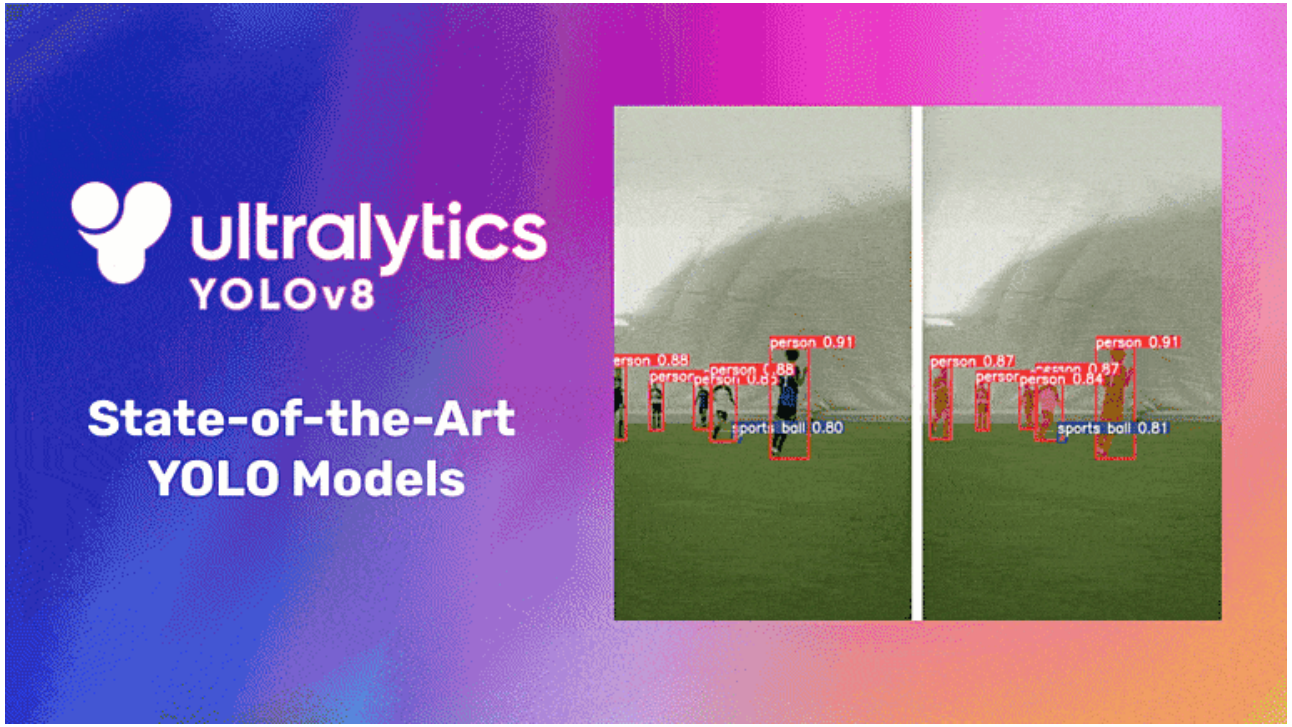
Homework 3



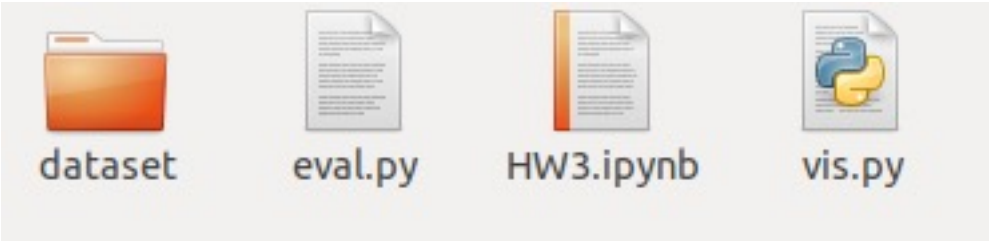
Problem 1 - Getting familiar with YOLOv8 (20%)
- Object detection
- Instance Segmentation
- Human Pose Estimation

Problem 2 – Custom YOLOv8 model training (20%)
- Train your YOLOv8 on the SportsMOT dataset

Problem 3 – Run tracking on SportsMOT dataset (60%)
- Baseline tracker implementation (optional)
- Run tracking – baseline or other opensource (10%)
- Run evaluation with MOTA and IDF1 (10%)
- Performance ranking (30%)
- Report (10%)



Code



What you got:

- Dataset – two videos from the SportsMOT dataset (1 for training and 1 for testing)
- eval.py – evaluation code to test your tracker's performance
- HW3.ipynb – where you write the code
- vis.py – visualization code to check your tracking result

What you submit:

- HW3.ipynb
- One report (in .pdf format)

Grading

1. Report (10%)
   - Experiments results
   - What did you attempt, and what is the results
   - Some screenshots of your tracking results
   - Any insights or thoughts
2. Ranking (30%)
   - This grade will be given based on your ranking of performance.
   - Performance ranking is based on $\sqrt{\text{MOTA} \times \text{IDF1}}$
   - How we give grade based on ranking is in next slide (but subject to change)

This grading is **subject to change**

```python
def grade_homework(MOT_scores):
    """
    Convert MOT performance scores into homework grades (0-30) based on a normal distribution.

    Args:
        MOT_scores (list or np.array): List of MOT performance scores (e.g., leaderboard positions or accuracy scores).

    Returns:
        np.array: Normalized homework scores ranging from 0 to 30.
    """
    # Convert scores to numpy array for processing
    MOT_scores = np.array(MOT_scores)

    # Normalize Kaggle scores to a standard normal distribution (mean=0, std=1)
    standardized_scores = (MOT_scores - np.mean(MOT_scores)) / np.std(MOT_scores)

    # Map the standard normal distribution scores to a scale of 0-30
    grades = norm.cdf(standardized_scores) * 30  # CDF maps to range [0, 1], then scale to [0, 30]

    max_grade = np.max(grades)
    rescaled_grades = (grades / max_grade) * 30

    return rescaled_grades
```

Some reference that might be useful

- Zhang, Yifu, et al. "Bytetrack: Multi-object tracking by associating every detection box." *European conference on computer vision*. Cham: Springer Nature Switzerland, 2022.
- Aharon, Nir, Roy Orfaig, and Ben-Zion Bobrovsky. "BoT-SORT: Robust associations multi-pedestrian tracking." *arXiv preprint arXiv:2206.14651* (2022).
- Yang, Fan, et al. "Hard to track objects with irregular motions and similar appearances? make it easier by buffering the matching space." *Proceedings of the IEEE/CVF winter conference on applications of computer vision*. 2023.
- Huang, Hsiang-Wei, et al. "Iterative scale-up expansioniou and deep features association for multi-object tracking in sports." *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2024.