Ravina G., Siddarth P, Devon F, Narendra P, Bonaventure R
Project Proposal

Project Overview

1. *Problem Statement*
    - *Initial Problem Statement:*
        - *i.* Tracking objects across cameras. Say if we have two cameras looking at pedestrians, can you figure out how to ID the same person from different perspectives?

Rewording the problem: After our meeting (10/23), we will change our approach to an identification problem. That is, classifying individuals that have been recognized by an image feed at an earlier occasion.

2. *Proposed Solution (Updated):*

Our new proposed solution to address identification would be broken into three portions namely- detection, training, and predictions. Initially, we would need to detect pedestrians which we still could use YOLO or TorchVision to accomplish such a task. Our next step would be to find a way to classify features to an individual. While approaching this problem, we thought of the idea of utilizing feature maps similar to those in a convolutional neural network in the hidden layers. However, those feature maps would be quite difficult to find per layer of hidden units (set of weights per depth). Each depth of hidden layers would also be processed from the previous layer and would lead to needless overhead of guessing useful features. Rather we decided to find a way to let the model figure out the important features. Thus, it seems logical to follow through with an auto-encoder, one that would seek to compress features (in the latent space) before using the de-encoder. We would need to use the encoder and de-encoder (the whole model) while training on a random open source dataset of individuals (provided below OR use a pre-trained model). When training the model it would be crucial to employ backpropagation to tweak our model to find important features. Afterwards, if an individual was to pass by, we would pass it through to the model. However, when making a prediction, we only use the feed forward portion and collect the vector produced by the latent space (compressed feature space). Essentially, this vector would produce a vector estimate of what it assumes to be that individual's feature set. We can save this vector in memory. Now, when we attempt to reclassify an individual, we would pass their image into the autoencoder. Ideally, the autoencoder will produce a vector

relatively similar to the original estimate. We can calculate say, the cosine distance between similar vectors to get a probability of whether this individual has been found before.

Initially, when we thought of this idea- we thought this was a novel solution. However, after some specific googling, we found a few papers at top venues that represent a very similar idea (and perhaps more optimized than the one we offered above).

Reference Link: https://arxiv.org/abs/1904.07223, https://arxiv.org/abs/1711.10295

* Since this problem has been addressed, we will seek to solve this problem in real time (unlike this paper) as well as with multiple cameras, rather than one. We offer this as an additional attribute to investigate. Furthermore, after reading their optimized solution- we could perhaps further build on their solution if we were to focus on different aspects (such as cars, animals, etc).

Dataset for training OR -- ** may use a pre-trained model.
https://drive.google.com/file/d/0B8-rUzbwVRk0c054eEozWG9COHM/view

3. *Research:*
    - <u>You Only Look Once Unified Real Time Detection System</u>
    - Base network identification for YOLOv1 was 45 fps (w/out Nvidia Titan intensive GPU batch training). → Increases for YoloV2,3
    - Evaluates images once via similar resolution training and test images and turns into a regression problem rather than a purely classification one.
    - N predictions per S x S box (S being number of input boxes). - Can be increased depending on dataset spatiality b/w cameras.
    - Limits number of predictions per box to one class- spatial constraint limits number of objects detected close to camera. (Depends on where the camera is placed)

    - <u>Single Shot MultiBox Detectors</u>
    - SSD for detection (SSD - initially performs @ 59 FPS) by eliminating box proposals.
    - SSD predicting boxes scales poorly when detecting with less feature maps and less default box shapes- i.e less default box ratios (page 10 SSD). Meaning, it performs better with more default boxes.
    - SSD outperforms YOLO in terms of accuracy w/good ratio of conv default boxes + multiple features maps + matching strategy
    - SSD needs significant amounts of training data to perform at the mAP in the paper, data augmentation is a potential solution (zoom in, zoom out)

    - <u>Comparisons</u>
    - It seems in our best interest to pick between newer versions of YOLO and SSD as they tend to outperform fast mask-rcnn, rcnn, and other DPM techniques within the domain of real time object detection (w/ FPS & accuracy).
    - Both do not use sliding window approaches or regional classification.
    - YOLO uses higher scale feature maps- performs worse for smaller objects, which may be beneficial for our task- if pedestrian tracking. SSD uses multiscale, may be better if tracking all objects as well (more fps as well). SSD apparently also performs better due to multiple ratio boxes (requires more data).

4. Data Collection

Business/Customers

We will be working with Lowe's to solve a real world problem. The solution using Computer Vision will help identify pedestrians from different perspectives using multiple cameras. It will also be helpful in other scenarios where one needs to identify a person for investigation purposes. We believe that similar solutions might exist in the market as computer vision is growing at such a rapid speed. We think that the audience for this project will mainly be organizations involved in investigations for theft identification or any other malpractices.

Academic Literature Review

We have referred three papers for our research. These papers mainly focus on object detection in the frame. One of the papers talks about the YOLO library which detects different types of objects and provides labels. The second paper states the other way to do object detection which is much faster than YOLO but leaves out the labelling part. The third paper focuses on re-identifying a person from multiple cameras.

Links to papers
YOLO: https://arxiv.org/abs/1506.02640
SSD: https://arxiv.org/abs/1512.02325
Person re-identification in multi-camera networks:
https://www.researchgate.net/publication/220669068_Person_re-identification_in_multi-camera_networks

Open Source

Both SSD and YOLO are open source projects and easily available. For testing and training purposes datasets are available in video and image formats.

Team Members

Group Members	**Technology Known**	**Additional Skills**
Ravina Gaikawad	Python, Java, C#, JavaScript, React-Native, SQL, MongoDB, AWS, NodeJS, Machine learning	Software Engineering Agile Methodology Full stack development Post Held: Software Engineer
Sidharth Panda	Java, Python, JavaScript, SQL, MongoDB, AWS, NodeJS, Machine learning, Data Science	Software Engineering Agile Methodology Front End and Back End development Post Held: Senior Software

		Engineer
Bonaventure Raj	Python, C++, C, Java, React, Deep Learning, Cloud Computing	Previous work: ML Intern & SWE Intern
Devon Faryadi	Python, Java, C, C++, C#, SAS, Tableau, DaVinci Resolve	Software Engineering Blockchain Development Video Editing
Narendra Pahuja	C#,Angular,Python,Java,Javascript, SQL,MongoDB,	Software Engineering. Agile Methodology. Full stack development. Post Held: Software Engineer