INFER - INtermediate representations for FuturE pRediction

Shashank Srikanth, Junaid Ahmed Ansari, R. Karnik Ram1, Sarthak Sharma1, J. Krishna Murthy, and K. Madhava Krishna

Presented By: Abhishek Kandoi, Aravind Mahadevan, Eric Megrabov, Sanjeev Sinha
Mentors: Nachiket Deo, Kaouther Messaoud, Mohan Trivedi
Project Overview

- **Objective** - Creating an input space to feed a predictive model for trajectory prediction
  - Takes in input representations which describe the history of the target and its environment

- **nuScenes dataset**
  - Contains annotations and samples for us to create these input generations
  - Each scene in the dataset contains frames, which we generate the inputs for the model

- **Training a convolutional LSTM model with a U-net structure over the nuScenes dataset**
  - Output from the model is a set of points in the global frame, which indicates future trajectory of a target in time - Compared with the ground truth

- **We want to compare how our approach works to a map-based representation approach**
  - Our approach uses semantic image info for learning rather than pixel info, as is the case normally

- **Novelty**: we are applying trajectory prediction on the ego vehicle (from which all of the data is taken) rather than other vehicles on a road
Project Overview - Pipeline
Semantic Segmentation: Overview

- Use intermediate representations to do trajectory prediction
- Critical portion is semantic segmentation
- Lidar and semantic segmentation together form intermediate representations
- Classes of each lidar point must be determined to separate different categories
Semantic Segmentation: Classes

- Trained using WideResNet-38 with In-Place ABN
- Mapillary Vistas dataset: 65 unique categories
- Employs transfer learning
- Classes:
  - Lane: Lane Marking - Crosswalk, Lane Marking - General
  - Road: Road
  - Obstacle: Building, Curb, Vegetation
  - Other Vehicles: Bicycle, Bus, Car, Caravan, Motorcycle, Other Vehicle, Trailer, Truck
  - Target Vehicle: Ego Vehicle
- Performed on all four available camera frames
Semantic Segmentation: Example

- Overall Segmentation
- Obstacle Segmentation
- Road Segmentation
- Lane Segmentation
Intermediate Representation Generation

- **Original Approach**
  - Initially decided to use semantic segmentation to create inputs for lane, road, obstacle representations
  - Instance segmentation for the other vehicle/target vehicle representation
- **Eventually realized this is a much harder way to classify points by object detection (for one object)**
  - Decided to use bounding boxes (in bird’s eye view)
    - Much simpler to obtain bounding box areas and see which lidar points are enclosed in these boxes
Intermediate Representation Generation Steps

1. Obtain All Lidar Points in a Frame
2. Obtain a mask to get all the lidar points that are inside the image plane.
Intermediate Representation Generation Steps

1. Project lidar points into the image plane of corresponding camera

2. ‘Paint’ each valid lidar point a semantic class
Intermediate Representation Generation (Cont.)

Project each painted lidar point into a bird’s eye view

Create each of the intermediate representation

- Lane
- Road
- Obstacles
- Other vehicles
- Target vehicle
Lidar points in back camera

Lidar points in front camera

Lidar points in front left camera

Lidar points in front right camera
Final point painted representation of scene
Challenges with Intermediate Representation (Cont.)

- Representation need to capture the change in objects over time.
- Transforming the lidar points to birds eye view (BEV) critical in order to capture the temporal nature of the representation.
- As a group tried two ways of projecting into BEV,
  - Using the bird’s eye view rendering code given by Nuscenes
  - Custom bird’s eye view rendering we created
Transform to BEV Attempt 1

- Used the BEV rendering code available on Nuscenes.
- Method transforms the lidar points in lidar frame to the ego vehicle frame and then performs the transformation to BEV.
- This method does not capture the temporal nature of the representation, specifically for the ego vehicle.
Transform to BEV Attempt 2

- Second attempt to render to BEV was to not transform to ego vehicle frame but to transform to BEV from global frame instead.
- At each frame, Nuscenes provides the ego pose of the vehicle and the translation gives the location of the ego vehicle in the global frame.
- Simple BEV transform proposed was to project all the points to be parallel to $z = 0$.
- With this representation, we can see that we can see the ego vehicle move over time unlike the previous representation.
- Issue with this representation is that it is not transferable across different scenes.
Transform to BEV Attempt 2
Transform to BEV Attempt 3

- Given an ego vehicle pose at a current time step, we transform the previous point cloud data and ego vehicle location with respect to the current ego vehicle pose.
- We subtract all the points in the previous point cloud data with the translation specified by the rotation.
- Final BEV representation because it did not suffer from the deficiencies the other two representations.
Intermediate Representation: Using Map

*obstacles channel includes walkways, traffic cones and barricades
Architecture

- Predicts VoI’s position on an occupancy grid given past intermediate representations
- Simple Encoder-Decoder model
- Connected by a convolutional LSTM to learn temporal dynamics
- *Skip connections* between corresponding encoder and decoder branch
- Input is sequence of intermediate representations
- Produces a single channel output occupancy grid
Training with Safety Loss

- Safety loss apart from MSE reconstruction loss (deviation from actual future state)
- Meant to penalize all predicted state of vehicles that lie in an obstacle cell
- Validation loss diverges a lot
- Worse performance than without
- Mentioned in paper but not included as part of code release

$$L_{safe} = ||O \odot \hat{F}||$$
Training with Safety Loss

\[ \mathcal{L}_{safe} = \|\mathcal{O} \circ \hat{F}\| \]
Current Progress

- We are finishing up the data loading pipeline and are generating the intermediate representation across all scenes.
- We are serializing the intermediate representations so that we do not need to go through this pipeline for each scene as it is extremely time consuming and will slow down training even more.
Current Workload Split

- Initial Training: Aravind, Abhishek
- Safety Loss Implementation: Sanjeev, Abhishek
- Semantic Segmentation: Aravind, Eric
- Instance Segmentation: Sanjeev
- Transformations: Aravind, Sanjeev
- Bounding-Box Work and Point Painting: Aravind, Eric
- Map Intermediate Representations: Abhishek
- Visualizations: Aravind, Sanjeev
Future Work

- Create data directory to organize intermediate representations
  - Taken from the nuScenes real dataset - Part 1
  - Start training on these intermediate representations - should take roughly 5 hours at the most
  - Have this done by June 8th

- Keep re-training/testing the model and obtain final results by June 10th
Thank you for your attention!

We would like to thank Professor Trivedi and his TAs, Nachiket and Kaouther! They helped us tremendously getting this far in the project, we couldn’t have done it without them.

Questions?