Vehicle Trajectory Prediction using Graph based agent interactions

Team Members:

Shrinidhi Venkatakrishnan
Srivaishnavi Sree Krishnan
Sutej Pramod Kulgod
Vishwas Nagesh Moolimani
Mentors

1. **Dr. Mohan Trivedi**  
   Member of LISA CVRR Lab  
   Professor, Electrical and Computer Engineering  
   UC San Diego

1. **Messaoud Kaouther**  
   Student, Inria  
   Member of LISA CVRR Lab

1. **Nachiket Deo**  
   TA, ECE 285  
   PhD student, LISA CVRR Lab  
   UC San Diego
For an autonomous vehicle, the ability to predict the possible future behaviors and trajectories of surrounding vehicles is essential to predict its own behavior and trajectory.

Towards this,

1. **CS-LSTM [1]:**
   Uses convolutional social pooling for capturing spatial dependencies and agent interaction.

2. **MHA-JAM [2]:**
   Uses multi-head attention on joint agents and map context to capture spatio-temporal interactions.

3. **GRIP [3] and GRIP ++ [4]:**
   Represent and utilize inter-object interaction using an undirected graph.
Objective

- To predict the future trajectories of vehicles in a scene by taking into account the spatio-temporal interactions of the objects in a scene.
- Use a graph based model to capture inter vehicle interactions and HD maps to capture information about the vehicle surroundings.
Expected goals

- Comparable performance to most state of the art trajectory prediction models by making use of:
  - Comprehensive Urban environment dataset - nuScenes.
  - Graph based implementation to obtain better spatial and temporal feature representation.
  - Multi modal trajectory prediction with probabilistic reasoning.
  - HD maps to capture information about the vehicle surroundings.

- Potential participation in the nuScenes trajectory prediction prediction challenge.
Novelty

- Implementing GRIP++ for nuScenes dataset.

- Incorporating a novel graph generation model which captures both spatial and temporal interactions between the vehicles in a given scene.

- Using an Autoencoder to learn the underlying distribution of future coordinates to achieve multimodal trajectory prediction.
Datasets

1. nuScenes dataset:
   a. It is a large self-driving car dataset captured using camera and Lidar sensors during urban driving in Boston, USA and Singapore.
   b. It is composed of 1000 scenes, each of 20 seconds records.
   c. Each scene record involve tracks hand-annotated at 2 Hz as well as high definition maps.

2. Apollo Scape dataset - Baseline dataset used in GRIP++:
   a. The trajectory dataset consists of camera-based images, LiDAR scanned point clouds, and manually annotated trajectories for over 1000km.
   b. It is collected under various lighting conditions and traffic densities in Beijing, China.
Current Progress: nuScenes Dataloader

- Train, validation and test data split is obtained using the official nuScenes prediction challenge devkit.
  32,186 observations - train set | 8560 observations - val set | 9041 observations - test set

- Each observation was grouped and processed scene-wise.

- All the past and the future frames annotations were obtained for every given instance from a scene.

Sample annotation for an observation from nuScenes
Current Progress: nuScenes Dataloader

- The below features are extracted per given frame considering the given instances in a scene:
  - Frame_ID
  - Instance_ID
  - Instance_type
  - X_coordinate and Y_coordinate of an instance
  - Heading_angle of an instance (calculated from consecutive xy coordinates)

- The extracted features are zero-centralized and converted into data structures as needed by the input of the model. The final input structure is as follows:

\[(N, C, T, V), (N, V, V), (N, 2)\]

- \(N\): number of datapoints
- \(C\): number of features for each datapoint (equal to 11)
- \(T\): \(t_h + t_f\) number of frames
- \(V\): Total number instances considered per scene (equal to 50)
Methodology: GRIP

GRIP: Graph-based Interaction-aware Trajectory Prediction
Methodology : GRIP++

Differences/Improvements over GRIP

- Takes velocity input.
- Considers both fixed and trainable graphs.
- Uses 3 blocks in the graph convolution model and adds batch normalization.
- Uses skip connections.
- Uses GRU networks.
Baseline-Graph construction

- Number of nodes: 4
- Threshold value: D-close
- 1’s represent the spatial edges

\[ A = \begin{bmatrix}
1 & 1 & 0 & 0 \\
1 & 1 & 1 & 0 \\
0 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 \\
\end{bmatrix} \]

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>B</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>C</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>D</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>

A \rightarrow B: D-close

B \rightarrow C: D-close

C \rightarrow D: D-close

D \rightarrow A: D-close
Current Progress: Improvements to Baseline

\[ A = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \quad \Rightarrow \quad \mathcal{A} = \begin{bmatrix} 0.5 & 0.5 & 0 & 0 \\ 0.33 & 0.33 & 0.33 & 0 \\ 0 & 0.33 & 0.33 & 0.33 \\ 0 & 0 & 0.5 & 0.5 \end{bmatrix} \]

\[ \mathcal{A}^2 = \begin{bmatrix} 0.41 & 0.41 & 0.16 & 0 \\ 0.27 & 0.38 & 0.22 & 0.11 \\ 0.11 & 0.22 & 0.38 & 0.27 \\ 0 & 0.16 & 0.41 & 0.41 \end{bmatrix} \]
Current Progress: Improvements to Baseline

Changes to Graph Convolution Block:

Baseline Graph Convolution Block

Graph Operation

n

Batch Normalization

Temporal Conv2D

n

n

n

n

Our Graph Convolution Block

Graph Operation

n'  n'  n'  n'  n'

Batch Normalization

Temporal Conv2D

Temporal Conv2D

Temporal Conv2D

Batch Normalization
**Metrics**

**Average Displacement Error (ADE):** Average L2 distance between ground truth and our prediction over all predicted time steps.

\[
ADE = \frac{\sum_{i=1}^{n} \sum_{t=1}^{T_{mat}} \left[ \left( \hat{x}_i^t - x_i^t \right)^2 + \left( \hat{y}_i^t - y_i^t \right)^2 \right]}{n(T_{pred} - (T_{obs} + 1))}
\]

**Final displacement error (FDE):** The distance between the predicted final destination and the true final destination at end of the prediction period $T_{pred}$. 
**Metrics**

**RMSE: Root Mean Square Error (RMSE):** is the standard deviation of the residuals (prediction errors). Residuals are a measure of how far from the regression line data points are; RMSE is a measure of how spread out these residuals are.

\[
RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (x_p - x_g)^2}
\]
Result:
Dataset: NuScenes
Epochs: 50

<table>
<thead>
<tr>
<th>Model Type</th>
<th>Mean ADE</th>
<th>Mean FDE</th>
<th>Mean RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>New model - Teacher Forcing</td>
<td>4.87</td>
<td>11.27</td>
<td>2.84</td>
</tr>
<tr>
<td>New model - No Teacher Forcing</td>
<td>4.49</td>
<td>10.33</td>
<td>2.51</td>
</tr>
<tr>
<td>Intermediate model - No</td>
<td>5.03</td>
<td>12.04</td>
<td>2.84</td>
</tr>
<tr>
<td>Teacher forcing</td>
<td>Intermediate model - Teacher Forcing</td>
<td>5.05</td>
<td>12.12</td>
</tr>
<tr>
<td>Baseline</td>
<td>4.55</td>
<td>10.40</td>
<td>2.58</td>
</tr>
</tbody>
</table>

Table 1: GRIP++ improvements
Detailed breakdown of individual workload

- **Shrinidhi & Sutej**
  - Improved the convolution block implementation to capture more temporal and spatial complexities.
  - Implemented a Graph construction method that captures both spatial and future temporal interactions.
  - Evaluation metrics were coded to compute the performance of the model.

- **Srivaishnavi & Vishwas**
  - Implemented the nuScenes dataloader and Lyft dataloader for GRIP++ and GRIP models.
  - Developed data preprocessing scripts to convert ‘observations’ given by the nuScenes challenge into data structures used by the models.
  - Ran baseline implementations of GRIP on Lyft dataset and GRIP++ on Apollo scape dataset.
  - Worked on the code for evaluation metrics to compute ADE, FDE and RMSE.
Future work for the next 2 weeks

- Employ a probabilistic representation of the output that considers multiple trajectories that the car can take.
- Incorporate the application of HD maps of nuScenes dataset to the input model and benchmark the same.
Milestones and timeline for the next 2 weeks

- Submit final report copy
- Complete full training on finalized model
- Tune model hyper parameters and submit report first draft
- Finalize on the multipath trajectory model
- Incorporating HD maps to the data

Dates:
- 4-Jun
- 6-Jun
- 8-Jun
- 10-Jun
- 12-Jun
- 14-Jun
- 16-Jun


Questions?