Deep Learning for Autonomous Driving

COMP4801 Presentation
Ho Chung Hin (3035567291)
Outline

• Introduction
• **Methodology** ← changes on existing model *(P2T)*
• Progress
• Conclusion
Introduction
Background

• **Task**
  - **Trajectory Prediction**
  - Where, When
  - What probability
  - Importance
    - Mimic human driver
    - Cope with uncertainty & interaction

• **Challenges**
  - Uncertainty & interaction
  - Real-time reaction
Dataset [1][2][3][4]

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Region</th>
<th>Label</th>
<th>HD Map</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>nuScenes</em></td>
<td>Singapore, US</td>
<td>23</td>
<td>Y</td>
</tr>
<tr>
<td>Stanford Drone</td>
<td>Stanford campus</td>
<td>4</td>
<td>N</td>
</tr>
<tr>
<td>PIT</td>
<td>Toronto</td>
<td>2</td>
<td>N</td>
</tr>
<tr>
<td>HEV-I</td>
<td>San Francisco Bay Area</td>
<td>8</td>
<td>N</td>
</tr>
</tbody>
</table>

- **Generalize across region**
  - Demographical: driving habit, races of the road users
  - Geographical: location, traffic density, weather
  - Others: traffic rules, road marking, lane structure

- **Handle vast variety of agents**

- **Utilize data available to modern cars**
Dataset: *nuScenes* [1]

• Prediction Task
  • Input: ≤2s history (any data)
  • Output: 6s predictions (2 FPS)
    • Format: x-y coordinates + probability

• Provided
  • Raw: image (front view), radar, lidar
  • **Annotation** (e.g. location, rotation angle, size)
  • **HD Map** (e.g. drivable area, stop line, traffic light)
Related Works [5]

- **Traditional Approaches**
  - Physics-based Model
  - Stochastic Model
  - Maneuver-based Model

- **Modern**: Interaction-aware Model
  - Pros: capture social context
  - Architecture: usually encoder-decoder
    - Encoder: spatial & temporal dimension
    - Decoder: usually with attention mechanism
Related Works [6][7][8][9]

<table>
<thead>
<tr>
<th>Model</th>
<th>Encoder (motion)</th>
<th>Encoder (lane)</th>
<th>Fusing</th>
<th>Pros</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Trajectron++</em></td>
<td>2 LSTM</td>
<td>2D CNN</td>
<td>2D CNN (x4)</td>
<td>Code Quality</td>
</tr>
<tr>
<td><em>AgentFormer</em></td>
<td>Transformer</td>
<td>2D CNN</td>
<td>Transformer</td>
<td>Reusable</td>
</tr>
<tr>
<td><em>GOHOME</em></td>
<td>1D CNN GRU</td>
<td>1D CNN GRU GCN</td>
<td>Cross-attention (motion to lane)</td>
<td>Scalable (range) Efficient (FLOPs)</td>
</tr>
<tr>
<td><em>P2T</em></td>
<td>NA</td>
<td>2D CNN (x2)</td>
<td>2D CNN (x6)</td>
<td>Open-sourced Best Submission</td>
</tr>
</tbody>
</table>
Objective

• Deliverable: *modified P2T*
  • Outdated: SOTA model has 10% lower error
  • Inefficient: sequential 2D CNNs (8 in total)

• **Goal**
  • Accuracy: 10% lower error
  • Efficiency: 50% lower ...  
    • Latency &
    • FLOPs (i.e. floating point operations per second)
Methodology
Agent Representation [6]

• **Dynamic Model**
  • Bicycle Model (e.g. vehicles)
    • Parameter: center of mass, front wheel steer angle...
  • Single Integrators (e.g. pedestrians)
    • Parameter: position, velocity...
  • Cons: difficult to do online estimation (i.e. real-time)

• **State**: mainly position (x-y coordinate)
  • Others: velocity, acceleration, yaw rate (change of heading angle)
  • Pros: simpler estimation, efficient computation
Reward Model

1. **Input**
2. **Feature Extraction Model**
3. **Reward Function**

- Depict possible route/endpoint

*Fig. 3 – architecture diagram of P2T*
Max-Ent RL

1. Learning Algorithm
2. Optimal Policy
3. Sampling (Plan)
   - Capture latent intent

Fig. 3 – architecture diagram of P2T
Trajectory Generator

1. Encoders

2. **Attention Mechanism**

3. Decoder (actual prediction)
   - Consider social-temporal feature

Fig. 3 – architecture diagram of P2T
Proposed Changes
Proposed Changes

1. Replace CNN with **GNN**
2. Utilize more map data
3. Use cross-agent attention

---

**Fig. 4** – graph representation

**Fig. 5** – highlighted reward model

**Reward Model**

- Static Scene: \( I \)
- Conv layers
- Track history: \( X \)
- Path rewards
- Goal rewards
Expectation - improve Efficiency & Accuracy [8]

<table>
<thead>
<tr>
<th>Model</th>
<th>K=6 minFDE</th>
<th>MR</th>
<th>#Param</th>
<th>FLOPs</th>
</tr>
</thead>
<tbody>
<tr>
<td>HOME</td>
<td>1.28</td>
<td>6.8</td>
<td>5.1M</td>
<td>4.8G</td>
</tr>
<tr>
<td>GNN-HOME</td>
<td>1.28</td>
<td>7.2</td>
<td>0.43M</td>
<td>0.81G</td>
</tr>
<tr>
<td>GOHOME</td>
<td>1.26</td>
<td>7.1</td>
<td>0.40M</td>
<td>0.09G</td>
</tr>
</tbody>
</table>

**Fig. 6 – comparison of CNN-based model & GNN-based model**

<table>
<thead>
<tr>
<th>Operation</th>
<th>CNN</th>
<th>GNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Context Information</td>
<td>2D convolutions</td>
<td>Vary (e.g. weighted sum in GCN)</td>
</tr>
<tr>
<td>Overlapped convolutions</td>
<td>Overlapped convolutions</td>
<td>Propagated along edges</td>
</tr>
<tr>
<td>Computation</td>
<td>Intensive (e.g. long range)</td>
<td>Efficient</td>
</tr>
<tr>
<td>Information Loss</td>
<td>1) resolution 2) implicit connectivity</td>
<td>Minimal</td>
</tr>
</tbody>
</table>

HOME: 2D CNN
GNN-HOME: 2D CNN + GNN
GOHOME: GCN
Expectation - improve Scalability [8]

**Fig. 7** – scale with output range

**Fig. 8** – scale with resolution
Implementation [8][10][11]

<table>
<thead>
<tr>
<th>Model</th>
<th>Encoder (motion)</th>
<th>Encoder (lane)</th>
<th>Fusing</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>LaneGCN</td>
<td>1D CNN</td>
<td>GCN</td>
<td>Cross-attention (x2) &amp; Self-attention (x2)</td>
<td>Complex interactions</td>
</tr>
<tr>
<td>GOHOME</td>
<td>1D CNN GRU</td>
<td>1D CNN GCN</td>
<td>Cross-attention (motion to lane)</td>
<td>Use of 1D CNN</td>
</tr>
<tr>
<td>PGP</td>
<td>GRU</td>
<td>GRU</td>
<td>Cross-attention (lane to motion) &amp; GAT</td>
<td>Less intuitive</td>
</tr>
</tbody>
</table>
Proposed Changes

1. Replace CNN with GNN
2. Utilize more **map data**
3. Use cross-agent attention

---

**Fig. 5** – highlight reward model

**Fig. 9** – map data (more context)
Proposed Changes

1. Replace CNN with GNN
2. Utilize more map data
3. Use **cross-agent attention**

Fig. 10 – cross-agent attention

Fig. 11 – highlighted trajectory generator
Expectation - improve Accuracy

• **More context** (e.g. traffic lights & car park areas)
• **Other agents' plan**

<table>
<thead>
<tr>
<th>Joint Socio-Temporal</th>
<th>ADE$_5$ ↓</th>
<th>FDE$_5$ ↓</th>
<th>ADE$_{10}$ ↓</th>
<th>FDE$_{10}$ ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ours w/o semantic map</td>
<td>1.97</td>
<td>4.21</td>
<td>1.58</td>
<td>3.14</td>
</tr>
<tr>
<td>Ours w/o joint latent</td>
<td>1.95</td>
<td>3.98</td>
<td>1.50</td>
<td>2.92</td>
</tr>
<tr>
<td>Ours (AgentFormer)</td>
<td><strong>1.86</strong></td>
<td><strong>3.89</strong></td>
<td><strong>1.45</strong></td>
<td><strong>2.86</strong></td>
</tr>
</tbody>
</table>

*Fig. 13 – ablation study from AgentFormer*

*Fig. 12 – map data & cross-agent attention*
Progress

- Implemented & created graph representation
- Removed dependency from trajectory generator
- GNN core implemented
  - **Unresolved**: dependency from Max-Ent RL module
  - **Plan**
    - Convert GNN output to compatible tensor; or
    - Remove/replace Max-Ent RL module
Conclusion

• Goal: **improve accuracy & efficiency**
  • Use GNN & graph
  • Use more map layer & attention
  • Backed by ablation studies

• Progress
  • **Implemented GNN & graph input**
  • Plan: deal with dependency from Max-Ent RL module
References


References


References
