Causal detector : classifying causal agents in a piedestrian trajectory

Causal representation to build more robust models

Author : Salim Cherkaoui

Data Science Project : 12 credits **Supervisor :** Yuejiang Liu

Abstract

As the use of machine learning models in motion forecasting for autonomous vehicles (AVs) becomes more widespread, it is crucial to ensure that these models make accurate and safe predictions. However, obtaining the large amount of data necessary to test a wide range of scenarios, including rare and challenging ones, is a difficult and expensive task. To address this issue, we try to come up with a new benchmark and causal labeling process for evaluating and improving the robustness of machine learning models.

1 Introduction

Machine learning models are increasingly prevalent in trajectory prediction and motion planning tasks for autonomous vehicles (AVs), therefore it is of a big importance to ensure that these models have robust and reliable predictions across a variety of scenarios. However, obtaining the data needed to evaluate and improve the robustness of these models can be difficult and expensive.

Waymo authors in the paper *CausalAgents: A Robustness Benchmark for Motion Forecasting* [1] propose a solution to this problem by perturbing existing data by removing certain agents from the scene, which allows them to evaluate and improve the models' robustness to spurious features. In this work, we will focus on the labelling process of these solution, and try to build a classifier that can label an agent as causal or non-causal to the ego agent in a scene of simulated pedestrian trajectories.

2 Related work

Waymo : In their work, Waymo [1] proposed perturbing existing data via agent deletions to evaluate and improve model robustness to spurious features. Since creating such modifications requires a high level of understanding of the scene and causal reasoning, they used human labelers to identify non-relevant agents. They defined *a non-causal agent as an agent whose deletion does not affect the ground truth trajectory of a given target agent* (we will also use this definition in this work). They created a robustness evaluation dataset that is composed of modified examples where they deleted all non-causal agents from each scenes, and they analyzed the model's behavior under different modifications, such as removing causal agents, a subset of non-causal agents, or stationary agents.

Waymo's labelling policy : Determining causality is a subjective task as human drivers may have different opinions on which agents in a scene affect their decisions. Therefore, to ensure the accuracy of their results, they aimed to be overly cautious and identify as many causal agents as possible. In ambiguous situations, they do not expect labelers to reason about chained causality relationships. For example, if the AV is driving behind a queue of 5 cars and the first car were to brake, it could eventually cause the car in front of the AV to brake. In this situation, they would only expect the labeler to identify the car directly in front as causal.

Causal annotation HONDA : In a related study [2] employs human labelers to gather causal annotations for the Honda Research Institute Driving Dataset and they evaluate the performance of an object detector across scenes with different causal characteristics. In contrast, Waymo's work utilizes causal labels to assess the robustness of trajectory prediction models on the Waymo Open Motion Dataset (WOMD) and they offer more detailed per-agent measurements of causality.

ROCK, Causal Inference Principles for Reasoning about Commonsense Causality : Commonsense causality reasoning (CCR) is the process of identifying potential causes and effects in natural language descriptions that are considered reasonable by a typical person. We'll use a causal labelling approach that is inspired by this causal framework [3].

3 Labeling and datageneration

Trajectories : We use the RVO2 Library which is an open-source implementation of an algorithm based on the optimal reciprocal collision avoidance (ORCA) [6] formulation. At runtime, each agent senses the environment independently and computes a collision-free motion in a common workspace. This formulation, optimal reciprocal collision avoidance (ORCA) [6], provides sufficient conditions for collision-free motion by letting each agent take half of the responsibility of avoiding pairwise collisions.

Labeling : We first tried using causal labeling per time step of a trajectory. Since RVO2 is computing each step of an agent using the position and velocity of its neighbours at a given time step (neighbours are agents in a predefined area surrounding the ego agent), we first

chose to define any neighbour as a causal agent, and non-neighbour as non-causal. We quickly dropped this definition for three main reasons:

- It does not follow the causal labeling process of Waymo that we eventually want to test our model on.
- This definition would be too simple and maybe not useful to work with : a causal agent would simply be an agent near to the ego agent.
- It leads to an all new reasoning problem : how should we reason about the indirectly causal effect of a certain agent on the ego agent.

We then decided to stick with the definition used by Waymo to define a causal-agent :

> *A non-causal agent as an agent whose deletion does not affect the ground truth trajectory of a given target agent*

To get the required data framework, RVO2 simulator is adapted. We simulate once a scene given a set of agents and a defined set of parameters for those agents. We then simulate one more time the trajectories for the same scene (initially), but this time removing one agent. Doing so, we can now compute the Average Distance Error (ADE) of the ego agent's trajectories between the factual and the counterfactual scene (Figure 3.1). This ADE represent the causal effect of the dropped agent on the ego agent.

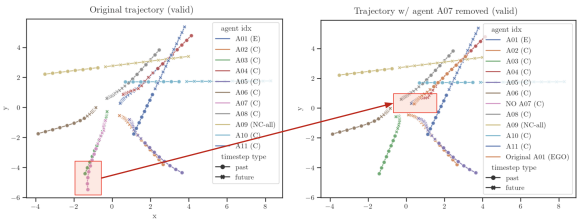

Figure 3.1: Factual trajectories of a scene with 11 agents at left, and the counterfactual associated to scene when removing agent n°7. The red rectangle in the left figure show the difference between the factual and counterfactual trjaectory of the ego agent.

Those ADEs are continuous labels. To be able to build a classifier and classify an agent as causal or not, we first labeled as non-causal any agent with an ADE of 0 and as causal otherwise. But this labels are too restrictive and maybe not very representative of realistic scenarios. Therefore, we decided to define 2 distinct thresholds for non-causal and causal agents.

$$\text{if } ADE < 0.001 \text{ then } label = 0 \text{ (NC)} \quad (1)$$

$$\text{if } ADE > 0.1 \text{ then } label = 1 \text{ (C)} \quad (2)$$

4 Methods implemented

4.1 Simple models

We try to build a very simplistic baseline. The following models only compare the trajectory of the ego agent with the one of the considered agent (the one we want to label).

- **MLP :** takes the trajectories of the two agents and output the chance that the agent is causal to the ego agent.
- **Relational Network :** using google's framework in their paper *A simple neural network module for relational reasoning* [4], we compared the two trajectories.
- **LSTM :** simple LSTM model, each cell takes as input the position of an agent at a certain time step, and takes also the hidden state of the previous cell. Those cells are first trained on next step predictions : each hidden state of a cell should be able to be used to predict the next position of the agent. Once this part of the training is done, we use all the hidden states from the cells as input of a Relational Network to classify the agent as causal or not.

4.2 Causal approach

A causal approach consists of comparing the factual scene with the counterfactual scene. To do so, we define the factual scene as the trajectories of all the agents of a given scene, and the counterfactual scene as the same trajectories but this time removing the trajectory of the considered agent (the one we want to label). We implement two diffetent model a **Causal-RN** and a **Causal-LSTM** using the same principles as above, but this time we compare the factual and the counterfactual scene, not only two agents.

4.3 STGAT-like classifier

We now build a more complex model using the encoder structure of the STGAT model [5].

4.3.1 Modeling Spatial-Temporal Interactions for Human Trajectory Prediction (STGAT)

The goal of the model [5] is to forecast the trajectories of the pedestrians involved in a scene. There are three components in the encoder: an LSTM-based pedestrian trajectory encoding module, a GAT-based module for modeling the spatial interactions, and an LSTM-based module for capturing the temporal correlations of interactions.

4.3.2 Classifier

We used the structure of the STGAT encoder and made some changes to be able to extract the encoded state of an agent in a given scene, and use it as input of a RN classifier (model's architecture in the Appendix Figure 7.1).

The main changes made on the STGAT encoder are the following :

- Every LSTM cells is pre-trained on next step prediction, whereas STGAT is training them on same step prediction.
- To encode the state of an agent, we take the hidden states of all the LSTM cells not only the last one.

5 Results and discussion

5.1 Results of each model

We evaluate each of the models on our training data using the mentioned thresholds to define the causality labels [Table 5.1] . Doing so, we get pretty good results on the simple models already, even more so than when using the causal approach or the STGAT-like model. Keep in mind that the MLP, RN and LSTM only use the trajectories of two agents. Which means that in the results got by those models, the interactions between the other agents in the scene are not taken into consideration. The results infer only on very simple features, most likely the distance between two agents, and their respected velocities. But this is not the problem we want to solve.

Models	Accuracy	F1-score	Specificity
MLP	0.872	0.871	0.878
RN	**0.891**	**0.889**	**0.906**
LSTM	0.872	0.869	0.896
C-RN	0.846	0.844	0.860
C-LSTM	0.859	0.858	0.864
STGAT-like	0.869	0.869	0.867

Table 1: Results of the different implemented model using the mentioned ADE thresholds for the labeling.

We can see in Figure 7.2, that the score predictions (output of the sigmoid in the classifier) is not really correlated to the ADE in itself. Of course we can see that extreme ADE tend to have a score that makes sens, but still, this is most likely due to the effect of a common cofounders.. This means that the model doesn't understand the causal effect in itself, but can see if an agent is a neighbour of the ego agent (using the RVO2 definition), which is fairly simple, and since most of neighbours have a causla effect on the ego agent, it therefore can classify the agent correctly (Figure 7.3 visualizes this point). Next part give a better proof and understanding of this claim.

5.2 Indirectly causal

To measure the point just mentioned, we now evaluate our models only on indirectly causal agents. Here, we define indirectly causal as the following :

> *An agent that is labeled non-causal with the ADE thresholds but has been a neighbour (RVO2 definiton) of the ego agent at least for one time step, is an indirectly causal agent.*

Label	Models	Accuracy
Indirectly causal	MLP	0.563
	RN	0.569
	LSTM	0.526
	C-RN	0.545
	C-LSTM	0.535
	STGAT-like	0.558

Table 2: Accuracy of the models when evaluating on the defined indirectly causal agents.

We can see [Table 5.2] that on these particular scenarios, the models perform very poorly, which was expected. Indirectly causal agents are agent that have an effect on an agent which by causation will have an effect on the ego agent. This means that to label correctly those agents, the models have to get a very good understanding of the scene and the different interactions. We try to solve that using the STGAT encoder which encodes the interaction between agents using attention, but this was not fruitful.

5.3 Discussion and limitations

The ADE labels come from removing an agent from a scene to compare it with the initial factual scene. This process is quite restrictive by itself. The model should be able to reason about the interactions between all the agents of a scene, but more than that. It should be able to understand the interactions between agents in the counterfactual scene too, but the data that we feed the model is the factual trajectories of the agents.

The causal approach is the one making the most sense, but the counterfactual scenario given to the model is only the scene without a certain agent. The ego agent's trajectory being unchanged. Doing so we are misleading the model, since the main trajectory in itself stays the same.

To be able to give a good counterfactual scenario to the model, we should feed it the real simulated counterfactual scene. but this have two main downsides. The first one being that it simply would be unrealistic, and the second is that it would be now too simple for the model to perform, since its job would simply be to compute the ADE between the factual and counterfactual trajectories.

Considering our labels, for the model to perform well, it should be able to reason about how the trajectory *would* change when you remove an agent. Which is quiet challenging with our current data framework. A good approach to tackle could be to use only a certain amount of time steps (observations) and to encode that to see if an agent will be causal in the next time steps. But this would need to adjust our labelling process accordingly, since for now, the ADEs are computed when removing an agent at the beginning of a scene.

6 References

References

[1] Rebecca Roelofs, Liting Sun, Ben Caine, Khaled S. Refaat, Ben Sapp, Scott Ettinger and Wei Chai (Waymo). CausalAgents: A Robustness Benchmark for Motion Forecasting

[2] Vasili Ramanishka, Yi-Ting Chen, Teruhisa Misu, and Kate Saenko. Toward driving scene understanding: A dataset for learning driver behavior and causal reasoning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 7699–7707, 2018.

[3] Jiayao Zhang, Hongming Zhang, Weijie J. Su, Dan Roth. ROCK: Causal Inference Principles for Reasoning about Commonsense Causality.

[4] Adam Santoro, David Raposo, David G.T. Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia and Timothy Lillicrap. A simple neural network module for relational reasoning

[5] Huang Yingfan, Huikun Bi, Zhaoxin Li and Tianlu Mao. STGAT: Modeling Spatial-Temporal Interactions for Human Trajectory Prediction

[6] Jur van den Berg, Stephen J. Guy, Jamie Snape, Ming C. Lin, and Dinesh Manocha Department of Computer Science, University of North Carolina at Chapel Hill. Optimal Reciprocal Collision Avoidance

7 Appendix

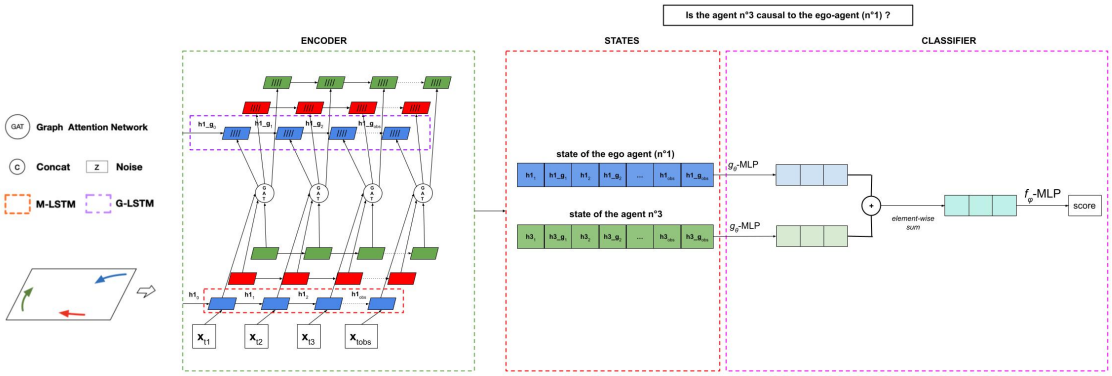

Figure 7.1: The architecture of our proposed STGAT-like classifier. The framework is based on the STGAT's seq2seq model and consists of 3 parts: Encoder, Intermediate State and the classifier. The Encoder module includes three components: 2 types of LSTMs and GAT. The Intermediate State encapsulates the spatial and temporal information of all observed trajectories, at every time step. The classifier compares the state of two agents using a relational network.

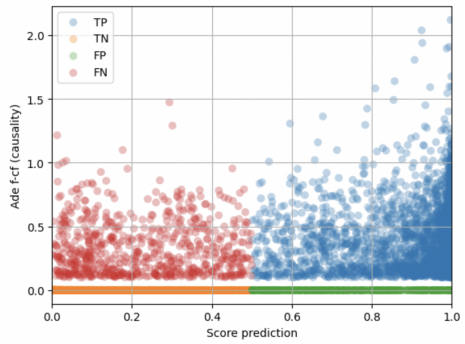

Figure 7.2: Score prediction (output of the classifier
for different scenarios and the associated ADE (causal effect of an agent on the ego agent).

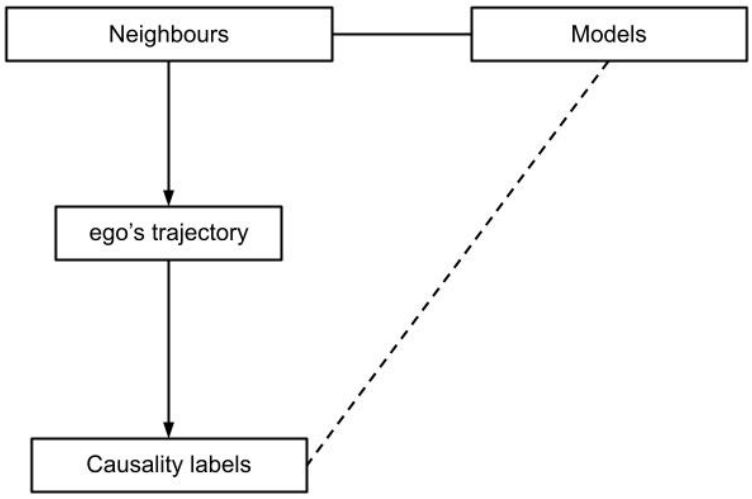

Figure 7.3: This how we assume that our models reason. They can detect if an agent is a neighbor of the ego agent (represented by the line between models and neighbors). Since neighbors influence the behavior of the ego agent, we in most of the case classify correctly a given agent. But what we want is too be able to reasoning directly on the causality of the agents (dashed line)