Motion Prediction for Self-Driving Needs a Metric Specific to Self-Driving

Neil Traft, Skanda Shridhar, Galen Clark Haynes
Uber ATG

***Abstract*— Commonly used metrics for motion prediction do not correlate well with a self-driving vehicle's system-level performance. The most common metrics are *average displacement error (ADE)* and *final displacement error (FDE)*, which omit many features, making them poor self-driving performance indicators. As both high-fidelity simulations and track testing can be resource-intensive, the use of prediction metrics better correlated with full-system behavior allows for swifter iteration cycles. We propose a framework for inventing and evaluating component-level metrics which are better correlated with system-level outcomes.**

I. Introduction

The task of detecting and predicting actors in a scene is an important part of most self-driving systems. Most approaches attempt this by training machine-learned models to predict trajectories or occupancy maps from sensor inputs. Building upon past object detection and tracking solutions, models are usually trained on variants of L2 distance or cross-entropy loss, and evaluated with metrics such as *average displacement error (ADE)*, *final displacement error (FDE)*, and *negative log likelihood (NLL)*.

When applied to self-driving, these have shortcomings. Consider Figure 1, which shows two scenarios with identical displacement error, however Fig. 1b is more critical, because the self-driving vehicle fails to anticipate an object entering its path. As a result, Rhinehart et al. [1] are able to show that two models which perform similarly on likelihood (R2P2 and PRECOG) have drastically different performance at the system level, in terms of collisions.

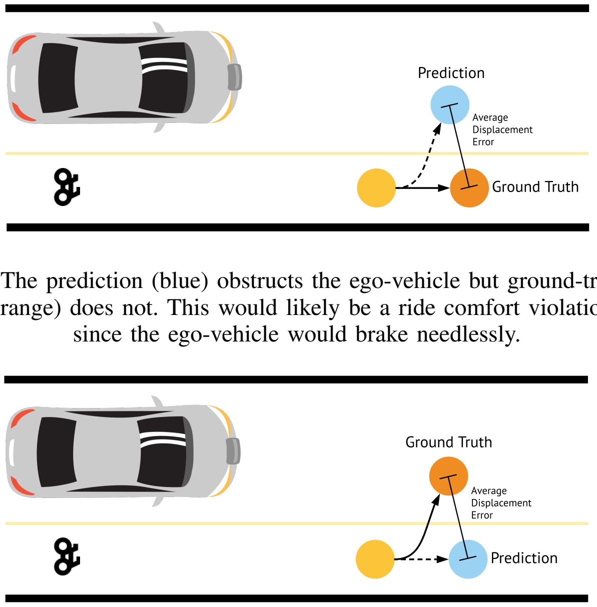
Gauging a model's impact on system-level behavior, therefore, requires simulation and track testing, both of which are expensive and require the presence of a fully-capable self-driving software stack. How shall we do a better job of evaluating perception/prediction at the component-level, independent of the overall system into which it is integrated?

II. Existing Motion Prediction Metrics

The metrics used for evaluating motion predictions today exhibit one or more flaws:

- No account of missed or false positive detections.
- No account of errors in shape or orientation.
- Only consider a single instant of time, or an average over all times.
- No disambiguation of which errors are relevant for driving (as in Fig. 1).

The most widely used metrics, ADE and FDE, were popularized by the TrajNet benchmark [2], [3] and are now used in many other benchmarks.



(a) The prediction (blue) obstructs the ego-vehicle but ground-truth (orange) does not. This would likely be a ride comfort violation since the ego-vehicle would brake needlessly.

(b) The ground-truth (orange) obstructs the ego-vehicle but the prediction (blue) does not. This case is concerning because it means the self-driving system does not foresee the need to brake.

Fig. 1: Two predictions with identical displacement error but very different system-level outcomes.

The nuScenes Prediction Challenge [4] and Argoverse Motion Forecasting Challenge [5] use the *minADE* variant, also referred to as "oracle error" [6], which takes the minimum error over the top k trajectories for a preset k. This incurs no penalty for situations such as Fig. 1a, and must be considered alongside regular ADE or FDE for a complete picture.

The Lyft Motion Prediction Competition [7] and other works [1], [8], [9] output a 2D probability distribution in the state space, and so are able to replace ADE with NLL.

All of these metrics carry common disadvantages.

- They only measure accuracy of a single point on an object, and do not account for orientation, shape, or relevance to the ego-vehicle.
- They present multiple objectives with no view as to how to trade them off against each other.
- They require *ground-truth association* (see below).

These benchmarks only evaluate motion prediction, rather than the joint tasks of detection, tracking, *and* prediction. Thus, users start with a perfect object track, and the label for each object is known. In the real self-driving task,

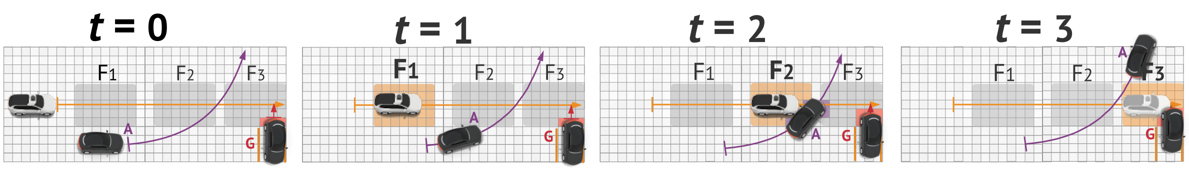


Fig. 2: At $t = 0$, a fixed ego-vehicle trajectory is shown as a sequence of its future *footprints*, F_1, F_2, and F_3. At $t = 1$, the ego-vehicle occupies F_1. At $t = 2$, it occupies F_2 and overlaps A's space. It cannot reach F_3 and encounter G at $t = 3$ (hence shown faded) since it must re-plan to stop for A, so A *blocks* G along this *specific* ego-vehicle trajectory.

we require *ground-truth association* algorithms: since the metric is defined w.r.t. a pair of actors, each predicted actor requires a ground-truth match. False positive and false negative detections are ignored and a prediction method can "cheat" if it only detects objects which are easier to predict (e.g., only near-range objects). An engineer must look at both prediction metrics *and* detection metrics and decide how to trade them off against each other. This is a major drawback.

III. Toward a Better Prediction Metric

We have sought to design a motion prediction metric which would better predict self-driving performance, and have gained some insights along the way.

A. How to Design a Better Metric?

To design component-level metrics which provide better signal of self-driving performance, we should ask ourselves, "What is the function of this component *toward the driving problem, specifically?*" Then we should design a metric which measures that function directly. Since there are many dimensions of system-level behavior (safety, ride comfort, reaction time, progress toward destination, etc.), we may well need to design separate metrics for each of these concerns.

The key idea: *A key function of predictions is to "block" the self-driving vehicle from colliding with a real-world object. However, we do not wish for predictions to block free space.* This concept is illustrated by Fig. 2. To measure this function, we:

1) Generate a set of ego-vehicle trajectories to approximate the full set of dynamically feasible maneuvers.
2) For each ego-vehicle trajectory,
   a) Compute the likelihood that any reachable objects along that trajectory are *not* "blocked" by predictions. *(safety-related)*
   b) Compute the likelihood that any reachable free space along that trajectory *is* "blocked" by predictions. *(comfort-related)*
3) Marginalize across all trajectories.

B. Meta-Metrics: How to Evaluate a Metric?

Now that we have designed a metric from the self-driving perspective, we seek evidence that this metric corresponds to system-level outcomes better than ADE or FDE.

Computing a standard correlation coefficient has not been particularly useful here. This is because the data do not have a simple functional relationship; system-level outcomes are determined by many factors. Consider a system-level metric like "unnecessary braking". The level of braking used to avoid an obstacle does not just depend on the obstacle's prediction, but also on the configuration of the scene, road geometry, vehicle dynamics, and so on. We hope that this variable will show some kind of co-dependence with our new metric, but we do not expect them to vary linearly or monotonically with respect to each other.

The key idea: *Start with a dataset of known prediction failures which lead to known system-level failures. If we have such failures, mixed in with a larger dataset of nominal driving, how quickly could an engineer discover these issues when looking through the data?*

Say we establish a dataset of safety-related failures known to be caused by prediction, where the self-driving vehicle is ≤ 2 meters from another object. These are mixed in with a larger dataset of nominal driving (which may contain other types of failures). Then, we rank all actors by our candidate "prediction safety" metric. We hope to see that the actors involved in the safety failures are ranked highly relative to all other instances. One way to quantify this would be,

> *Signal-to-Noise Ratio (SNR):* Given this ordering of actors across all simulations, what fraction of the top N actors correspond to the known safety concerns?

If a metric performs well on this stat, then an engineer can use this metric to efficiently disambiguate the prediction errors that impact driving performance from those that don't.

IV. Conclusion

In this paper we have surveyed the field of motion prediction metrics and outlined why none of them are expected to correlate well with system-level outcomes. We then provide a high-level outline of a metric approach better suited for self-driving, as well as a process for how to evaluate such metrics in a quantitative way.

References

[1] N. Rhinehart, R. McAllister, K. Kitani, and S. Levine, "Precog: Prediction conditioned on goals in visual multi-agent settings," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 2821–2830.

[2] A. Sadeghian, V. Kosaraju, A. Gupta, S. Savarese, and A. Alahi, "Trajnet: Towards a benchmark for human trajectory prediction," *arXiv preprint*, 2018.

[3] S. Becker, R. Hug, W. Hubner, and M. Arens, "Red: A simple but effective baseline predictor for the trajnet benchmark," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 0–0.

[4] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. (2020) nuscenes prediction task. [Online]. Available: https://www.nuscenes.org/prediction

[5] M.-F. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan *et al.* (2019) Argoverse motion forecasting competition. [Online]. Available: https://evalai.cloudcv.org/web/challenges/challenge-page/454/evaluation

[6] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. Torr, and M. Chandraker, "Desire: Distant future prediction in dynamic scenes with interacting agents," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 336–345.

[7] J. Houston, G. Zuidhof, L. Bergamini, Y. Ye, A. Jain, S. Omari, V. Iglovikov, and P. Ondruska. (2020) Lyft motion prediction for autonomous vehicles. [Online]. Available: https://github.com/lyft/l5kit/blob/master/competition.md

[8] J. Schulz, C. Hubmann, J. Löchner, and D. Burschka, "Interaction-aware probabilistic behavior prediction in urban environments," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 3999–4006.

[9] Y. Chai, B. Sapp, M. Bansal, and D. Anguelov, "Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction," *arXiv preprint arXiv:1910.05449*, 2019.