Forecasting from LiDAR via Future Object Detection

Neehar Peri1* Jonathon Luiten1,2 Mengtian Li1 Aljoša Ošep1,3 Laura Leal-Taixe3,4 Deva Ramanan1,4

1Carnegie Mellon University 2RWTH Aachen University 3TUM Munich 4Argo AI
{nperi, jluiten, mengtial, aosep, deva}@andrew.cmu.edu, leal.taixe@tum.de

Abstract

Object detection and forecasting are fundamental components of embodied perception. These two problems, however, are largely studied in isolation by the community. In this paper, we propose an end-to-end approach for detection and motion forecasting based on raw sensor measurement as opposed to ground truth tracks. Instead of predicting the current frame locations and forecasting forward in time, we directly predict future object locations and backcast to determine where each trajectory began. Our approach not only improves overall accuracy compared to other modular or end-to-end baselines, it also prompts us to rethink the role of explicit tracking for embodied perception. Additionally, by linking future and current locations in a many-to-one manner, our approach is able to reason about multiple futures, a capability that was previously considered difficult for end-to-end approaches. We conduct extensive experiments on the popular nuScenes dataset and demonstrate the empirical effectiveness of our approach. In addition, we investigate the appropriateness of reusing standard forecasting metrics for an end-to-end setup, and find a number of limitations which allow us to build simple baselines to game these metrics. We address this issue with a novel set of joint forecasting and detection metrics that extend the commonly used AP metrics from the detection community to measuring forecasting accuracy. Our code is available on GitHub.

1. Introduction

Object detection and forecasting are fundamental components of embodied perception that are often studied independently. In this paper we rethink the methods and metrics for trajectory forecasting from LiDAR sensor data. Trajectory forecasting is a critical perception task for autonomous robot navigation, thus building meaningful evaluation metrics and robust methods is of utmost importance.

Traditional trajectory forecasting methods [6, 9, 47] directly predict future object locations and backcast to determine where each trajectory began. Our approach not only improves overall accuracy compared to other modular or end-to-end baselines, it also prompts us to rethink the role of explicit tracking for embodied perception. Additionally, by linking future and current locations in a many-to-one manner, our approach is able to reason about multiple futures, a capability that was previously considered difficult for end-to-end approaches. We conduct extensive experiments on the popular nuScenes dataset and demonstrate the empirical effectiveness of our approach. In addition, we investigate the appropriateness of reusing standard forecasting metrics for an end-to-end setup, and find a number of limitations which allow us to build simple baselines to game these metrics. We address this issue with a novel set of joint forecasting and detection metrics that extend the commonly used AP metrics from the detection community to measuring forecasting accuracy. Our code is available on GitHub.

Figure 1. (a) Current stage-wise methods independently address the problems of detection, tracking, and forecasting, allowing for compounding errors in the full pipeline. Each sub-module incorrectly assumes that its input will be perfect, leading to further integration errors. In contrast to current forecasting methods that use object tracks as input, end-to-end forecasting directly from LiDAR sensory data (b) streamlines forecasting pipelines. To this end, we propose FutureDet (c), an end-to-end model capable of forecasting multiple-future trajectories directly from LiDAR via future object detection. We show that our end-to-end pipeline improves upon state-of-the-art three-stage and end-to-end methods.

...tect [44–46] and track [23, 48, 50] objects in 3D LiDAR scans to obtain past trajectories (Fig. 1a). These can be used in conjunction with auto-regressive forecasting methods [1, 20, 22, 43] to estimate the future actions of surrounding agents. Recent efforts [33, 37, 51] streamline such multi-stage perception stacks and train multi-task neural networks to jointly detect, track and forecast object positions directly from raw sensor data (Fig. 1b). However, such end-to-end approaches tend to predict only a single future trajectory for each object, not accounting for future uncertainty. This is not surprising as estimating multiple futures is a significant challenge in forecasting, requiring machinery such as multiple-choice-loss [5] or generative models [2, 10, 11, 20, 22, 25, 29, 42].

We rethink the forecasting task and propose FutureDet, an approach that reframes forecasting as the task of fu-
true object detection (Fig. 1c). Importantly, existing detectors [27, 53, 56] already learn to predict heatmaps that capture distributions over possible object locations. We repurpose this machinery to represent possible future object states. To this end, we encode an accumulated sequence of past raw LiDAR scans using standard backbones for 3D LiDAR-based object detection and train our network to (i) detect objects multiple timesteps into the future and (ii) estimate trajectories for these future detections back in time (i.e., back-cast) to the current frame. By matching backcasted future detections to current detections in a many-to-one manner, our approach can represent a distribution over multiple plausible future states. Our extensive evaluation on the large-scale nuScenes [6] dataset for trajectory forecasting reveals that our proposed FutureDet outperforms state-of-the-art methods, without requiring object tracks or HD-maps as inputs to the model. We posit that tracking may emerge from our network (since tracking objects from accumulated past LiDAR scans may make them easier to forecast), similar to the emergence of tracking and forecasting in streaming perception [31].

Furthermore, we investigate the utility of current metrics [33, 49] for evaluating forecasting directly from raw LiDAR data. We find that existing metrics are not well suited for the task of joint detection and forecasting, allowing them to be gamed by trivial forecasters. Current metrics for end-to-end LiDAR forecasting adapt trajectory-based forecasting metrics, such as average/final displacement error (ADE/FDE). These metrics were designed for evaluating forecasting in a setting where perfect tracks are given as input, and objects don’t have to be detected. However, these metrics don’t adapt well to the end-to-end setting. We demonstrate that such metrics can be gamed by baselines that simply rank all stationary objects (which are trivially easy to forecast) with high confidence, dramatically outperforming all prior art. Moreover, these evaluation metrics detach two inherently inter-connected tasks of detection and forecasting. Consequently, they do not penalize false forecasts, i.e., forecasts that do not actually belong to any objects. In this sense, the end-to-end setup and evaluation is more realistic.

To address these short-comings, we rethink the evaluation procedure for joint object detection and forecasting directly from sensor data. Our key insight is that the versatile average precision (AP) metric, a gold standard for assessing object detection performance, can be generalized to the task of joint detection and forecasting. The key feature of our novel forecasting mAP is that a forecast is correct only if the object is both correctly detected and forecasted. Our forecasting mAP is then calculated by simply using the machinery of AP, but using this joint detection and forecasting definition of a true positive. Furthermore, our forecasting mAP can be extended to evaluating multiple-future forecasts for each object by simply evaluating w.r.t. the top-K most confident forecasts per-detection. Our metric appropriately adapts forecasting metrics for end-to-end evaluation: forecasting mAP jointly assesses forecasting and detection, penalizing both missed forecasts as well as false forecasts. It assesses forecasting performance on the full set of object detections and embraces the inherent multi-future nature of forecasting.

Contributions: We (i) repurpose object detectors for the task of end-to-end trajectory forecasting and propose a model that can predict multiple forecasts for each current detection, (ii) rethink trajectory forecasting evaluation and show that detection and forecasting can be jointly evaluated using a generalization of well-accepted object detection metrics, and (iii) thoroughly analyze the performance of our model on the challenging nuScenes dataset [7], showing that it outperforms both previous end-to-end trainable methods and more traditional multi-stage approaches.

2. Related Work

Object Detection and Tracking. Due to recent advances in supervised deep learning [26] and community efforts in dataset collection and benchmarking [4, 6, 12, 16, 47], the research community has witnessed rapid improvement in LiDAR-based 3D object detection [27, 39, 44, 45], tracking [48, 53], and segmentation [3, 4, 54]. Several methods [44, 45] follow a well-established two-stage object detection pipeline using point-cloud encoder backbones and a 3D variant of a region proposal network [40], or detect objects as points, followed by classification and bounding box regression [53]. Due to the sparsity of LiDAR point clouds, recent methods accumulate multiple scans over time to improve object detection [27, 53] and LiDAR panoptic segmentation performance [3]. To understand how trajectories of detected objects evolve over time, multi-object tracking methods associate detections using Kalman filters [48], learned object descriptors [14, 50] or regress frame-to-frame offsets [53, 55], typically followed by greedy or combinatorial optimization to resolve ambiguous data association. The latter approach can be interpreted as a single frame forecast for track association. However, autonomous vehicles must account for likely future positions of surrounding agents at longer temporal horizons to safely navigate and avoid collisions in dynamic environments.

Trajectory Forecasting. Vision-based trajectory forecasting has been posed as the task of predicting future trajectories of agents, given perfect past trajectories and a top-down image (recorded e.g. using drones) as input [30, 38, 41]. Early physics-based models [21] explicitly model agent-agent and agent-environment interactions and have been successfully used to enhance object trackers [28, 52]. Recent methods use auto-regressive data-driven models that leverage recurrent neural networks (RNNs) and encoder-
decoder-based architectures to encode the past trajectory and estimate its evolution in future frames [1]. To deal with the inherent multi-modal nature of the problem, several methods leverage generative models [18, 24] to learn a distribution over future trajectories [2, 10, 11, 20, 22, 25, 29, 42]. However, these methods and benchmarks tackle forecasting in idealized scenarios, in which the entire visual scene is directly observed, and perfect input trajectories are provided. Both are unrealistic assumptions in automotive and robotics applications. Due to the importance of forecasting for automotive path planning, recent large-scale automotive benchmarks [6, 9, 47] explicitly focus on the trajectory forecasting task. Similar to vision-based methods, these benchmarks pose the forecasting problem as trajectory estimation given past trajectories and a high-definition map of the environment. These benchmarks have facilitated a wide suite of HD-map-based forecasting methods, which either represent the HD-map as rasterized images [8, 17], graphs or vectors [15, 19, 32]. However, both algorithms and the evaluation protocol make the unrealistic assumption that detection and tracking outputs are perfect.

**Forecasting from Sensor Data.** Prior methods [33, 37, 51] tackle object detection, tracking, and forecasting jointly by training a single convolutional neural network in a multi-task fashion from accumulated stacks of LiDAR sweeps. Alternatively, [49] directly forecasts future LiDAR scans and leverages off-the-shelf LiDAR object detectors to detect objects in these forecasted scans. We believe that this approach of end-to-end joint detection and forecasting is a step in the right direction. However, none of the aforementioned end-to-end methods are able to reason about multiple future trajectories. To embrace the inherently uncertain future, we present an end-to-end forecasting model (Sec. 4) that simultaneously detects objects in the current and future timesteps given a history of LiDAR scans, anchoring multiple possible future detections to current scan detections. This approach not only outperforms the aforementioned methods w.r.t. forecasting accuracy but also allows for multiple future interpretations. Finally, we observe that ad-hoc adaptations of forecasting metrics [33, 37, 51] do not appropriately characterize certain types of forecasting errors. As a remedy, we propose a generalization of the average precision (AP) [13] metric for joint detection and forecasting in Sec. 3. Note that our adoption of AP is also inspired by the work of streaming perception [31], where AP is used to measure the joint performance of 2D object detection, tracking, and short-term forecasting without considering multiple futures.

### 3. Rethinking Forecasting Evaluation

Since we are tackling the task of forecasting future positions of cars directly from LiDAR scans, we assume an observed sequence of past LiDAR sensor data, up to the most recent observation $S_{t_{obs}}$ at time $t_{obs}$, as input. We pose joint object detection and forecasting as the task of estimating a set of object locations (parametrized as 3D cuboids) in the current scan $S_{t_{obs}}$ as well as their future trajectory continuations in the future, unobserved scans i.e., $\{S_{t}, t \in [t_{obs}+1, \ldots, T]\}$.

#### 3.1. Preliminaries

Prior work [1, 20, 22, 30, 38, 41, 43] presents forecasting as the task of estimating the “correct” continuation of a given ground-truth track. In particular, given past trajectory observations $X_i = \{(x_i^t, y_i^t) \in \mathbb{R}^2, t = 1, \ldots, t_{obs}\}$, the task is to estimate future positions $Y_i = \{(x_i^t, y_i^t) \in \mathbb{R}^2, t = t_{obs}+1, \ldots, t_T\}$ for all agents present in the scene. This formalization is also adopted by recent automotive forecasting benchmarks [6, 9, 47]. However, as the ego-vehicle is moving, methods are given high-definition (HD) maps of the surrounding environment and ego-vehicle positions to account for the geometry of the surroundings. First, we discuss existing metrics for forecasting evaluation.

**ADE and FDE.** Average displacement error (ADE) and final displacement error (FDE) are commonly used evaluation metrics for assessing trajectory prediction. Both are measured as the L2 distance between model predictions $\{Y_i\}$ and ground truth trajectories $\{G_i\}$. To account for the inherent uncertainty in trajectory continuation, methods are evaluated over the set of top-K model predictions (w.r.t. the confidence score of each predicted forecast). These metrics assume that the set of true positives are the same for all methods. This assumption does not hold when comparing end-to-end methods, which can produce different sets of true positives, making comparison unreliable.

**Miss Rate.** If the final displacement error between a ground-truth trajectory and a prediction is above a center distance threshold, we count the forecast as a miss (similarly evaluated w.r.t. the set of top-K predictions). This metric evaluates the proportion of misses over all forecasts in a scene.

**ADE/FDE w.r.t. Recall.** The standard forecasting setup allows us to build models in isolation from other factors and has sparked rapid progress in this field of research [1, 10, 11, 20, 22, 43]. However, the standard assumption of having perfect input trajectories is not feasible in practice as it critically depends on perfect object tracks as inputs, which are nearly impossible to obtain in practice. To this end, [33, 37, 51] study end-to-end trajectory forecasting directly from raw sensor data, and propose an evaluation setup for end-to-end forecasting models using the aforementioned ADE and FDE at fixed recall thresholds, i.e., ADE/FDE at 60% or 90%. This evaluation setting has two major short-comings:
Evaluated only on a subset of matched detections. A large number of agents are not moving and prediction of their future positions is trivial. Models that rank stationary objects higher than moving objects can obtain higher forecasting performance by specializing on trivial predictions. We provide empirical evidence for this in Sec. 5.1 and show that such recall-based metrics can be “gamed” using a simple constant position prediction model.

No penalty for false positives. The current evaluation protocol detaches the inter-linked tasks of detection and forecasting. As a consequence, models can predict an arbitrary number of forecasts that are not anchored to any detections. In other words, this approach does not penalize false forecasts, i.e., forecasts not anchored to any detection, and missed forecasts as commonly characterized by the miss rate.

3.2. Average Precision is All You Need

Average Precision (AP). AP is defined as the area under the precision-recall curve [13], commonly averaged over multiple spatial overlap thresholds [34]. To compute AP we first determine the set of true positives (TP) and false positives (FP) to evaluate precision and recall. In standard object detection, TP's are considered to be successful matches between model predictions and ground-truth, typically determined based on 2D/3D intersection-over-union (IoU) [13] or distance from the object center [6] in the reference image or LiDAR point cloud, respectively. We can extend AP for joint detection and forecasting by (a) evaluating detection accuracy on the current frame or (b) evaluating detection accuracy T seconds into the future. However, (a) completely ignores forecasts and (b) doesn’t ensure that future trajectories are correctly associated to the right current detection.

Forecasting Average Precision (FAP). For the task of joint detection and forecasting, all future forecasts need to be anchored to objects, present (and detected) in $S_{t_{obs}}$. A robust metric must correctly penalize trajectories with correct first frame detections and incorrect forecasts (false forecasts), and trajectories with incorrect first frame detections (missed forecasts).

To characterize both types of errors, we define a true positive with reference to the current frame $t_{obs}$ if there is a positive match in both the current timestamp ($t_{obs}$) and the future (final) timestep $t_{obs} + T$. Otherwise, a forecast is considered to be a false positive. A successful match in the current frame is determined based on the distance from the center, averaged over distance thresholds of $\{0.5, 1, 2, 4\}$m [7]. Similarly, a successful match in the final timestep is determined based on the distance from object center, averaged over distance thresholds of $\{1, 2, 4, 8\}$m respectively. In contrast to ADE @ Recall % and FDE @ Recall %, this evaluation setting (i) takes all detections (not just true positives) into account, and (ii) penalizes missed forecasts (typically characterized by the miss-rate).

Forecasting Mean Average Precision (mFAP). Forecasting AP is evaluated on the full set of detections and cannot be “gamed” by a simple constant position model. However, we note that the data itself is imbalanced: over 60% of cars in the nuScenes dataset are parked, and are therefore stationary. To this end, we define three sub-classes: static car, linearly moving car and non-linearly moving car. Computing sub-class $FAP_i$ can be difficult; we do not require forecasts to output sub-class labels, but assume all ground-truth objects have sub-class labels. We follow the protocol for large-vs-small object subclass evaluation from COCO [35], described further in the appendix. We then evaluate $mFAP$ as $\frac{1}{3}(FAP_{stat} + FAP_{lin} + FAP_{non-lin})$ to ensure our metric cannot be “gamed” by trivial forecasters, as well as discuss fine-grained analysis on the three cases separately. Similarly, $mDAP_{det}$ is evaluated as the average $DAP_{det}$ over the three sub-classes.

Metrics: Embracing Multiple Futures. As described, $mDAP_f$ would be suitable for evaluating forecasting for scenes with multiple future ground truth trajectories. However, this is not feasible in the practice when forecasting directly from historical sensor data. To this end, we adopt a top-K based forecasting evaluation [30, 38, 41], that does not penalize models for hypothesizing possible future trajectories anchored from a single detection. In particular, we first match predictions to ground-truth detections in $t_{obs}$ and take the top-K highest ranked forecasts for each detection. Based on this set, we determine the best-matching forecast in terms of FDE and evaluate $DAP_f$ as explained above.

4. Forecasting as Future Object Detection

FutureDet addresses the forecasting problem by predicting the future locations of objects observed at $t_{obs}$. We can repurpose existing LiDAR detectors to predict object locations for $T$ future (unobserved) LiDAR scans, for which ground truth supervision is given. We first describe our method and discuss our implementation based on the recently proposed CenterPoint LiDAR detector. [53]

Future object detection and forecasting are related tasks. Forecasting requires predicting consistent trajectories in every frame between the current frame and $T$ future frames. To estimate forecasts from future detections we train our network to additionally estimate velocity offset vectors for every future detection. We do so for all frames between the current timestep and the final future timestep where the future detection occurs.

Backcasting vs. Forecasting. Fast and Furious [37] proposes a similar architecture that forecasts position offsets into the future directly from current-frame detections. Our
method considers the inverse setup where we detect in both the current and future frames and predict offsets back in time. We posit that future object detection requires the network to learn forecasted feature representations [36], directly optimizing for future object positions. Our experimental evaluation and visual inspection confirm this intuition (Sec. 3).

**Method: Embracing Multiple Futures.** The task of forecasting is inherently ambiguous: while there are many plausible outcomes given an input trajectory, only one future is realized for training supervision and evaluation. Traditional forecasting methods and benchmarks facilitate multiple future predictions via top-K based evaluation, leveraging multiple-choice-loss [5] and generative models [2, 10, 11, 20, 22, 25, 29, 42] to learn a (possibly multi-modal [10, 11]) distribution over future trajectories. \textit{FutureDet} allows for natural multi-future forecasting to emerge. We first point out that detection networks can be easily repurposed for future detection by giving target bounding boxes \( T \) seconds into the future. Since future objects are detected independently from current-frame detections, we posit that the network will produce multiple future detections for every object in the scene, effectively placing “multiple bets” where the objects may end up in the future. As all future detections are modeled by Gaussian heat maps, we implicitly obtain a multi-modal distribution over possible future locations (see Fig. 2).

**Matching Multiple Forecasts.** The task of forecasting requires all trajectories to be anchored to the set of object detections in the current (observed) LiDAR scan. For every future detection \( i \), we backcast and compute the distance to each detection \( j \) from the previous timestep. For each \( i \), we pick the best \( j \) (allowing for many-to-one matching). This framework naturally allows potentially multiple future forecasts to belong to each current timestep detection.

**Ranking Multiple Forecasts.** For all forecasts anchored at a single detection, we rank trajectories according to their forecasting score, derived using the detection confidence score of the last detection in a predicted trajectory. As shown in Table 2, we find there is a slight increase in performance between \( K = 1 \) and \( K = 5 \), indicating that better ranking strategies can further improve \textit{FutureDet}.

**Implementation.** We train CenterPoint to detect objects in future scans. The underlying detection network simply thinks it’s finding \( T \) times as many object classes (e.g., \textit{cars} and \textit{future-cars}) with additional regression offsets (analogous to existing velocity regressors). In addition, we repurpose the ground truth sampling (\textit{a.k.a.} copy-paste) augmentation [57] to increase the diversity of training trajectories. This provides considerable improvement in linear and non-linear forecasting performance. We use the PyTorch toolbox to train all models for 20 epochs with the Adam optimizer and a one-cycle learning rate scheduler.

**CenterPoint is already a one-frame forecaster.** It detects objects and predicts one-frame future velocity vectors that are used as cues for tracking. It does this by accumulating \( T \) previous LiDAR scans and encodes the accumulated point cloud sequences using a VoxelNet [56] backbone. Such a tracker could be used as input to auto-regressive forecasting methods, \textit{e.g.}, [43], however, we argue that we can use such a spatiotemporal representation to directly forecast.

**CenterPoint models object locations as Gaussians.** It does so by producing a 2D bird’s-eye-view (BEV) heatmap, which models the continuous likelihood of detections at each point in the BEV space. Detections are obtained by finding local maxima in these heatmaps via non-maximum suppression (NMS). By reusing this representation for future detection, our detection heatmaps are effectively a forecast of a continuous likelihood field for the locations of objects. This continuous field naturally encodes the uncer-
tainty of future detections, accounts for multi-modality, and provides a continuous representation for forecasting.

5. Experimental Evaluation

We conduct our experimental analysis on the nuScenes [6] dataset. As we are tackling end-to-end forecasting from sensor data, we do not follow the established evaluation protocol that provides ground-truth trajectories and HD maps as input (as explained in Sec. 3.1). First, we perform breakdown analysis of evaluation metrics proposed in [33] and our forecasting mAP by analyzing the performance of a simple constant position model (Sec. 5.1). After verifying that our proposed evaluation setting is not easily “gamable”, as discussed in Sec. 3.1, we thoroughly ablate our model and compare its performance to other state-of-the-art methods (Sec. 5.2).

Repurposing NuScenes Tracking Dataset. nuScenes [6] recently introduced a large-scale multi-modal dataset recorded in Boston and Singapore. It provides 1000 twenty-second logs that are fully annotated with 3D bounding boxes. This work explicitly focuses on forecasting based on LiDAR data, obtained with a 32 beam LiDAR sensor recorded at 20 Hz, covering 360-degree view. We follow the official protocol and evaluate forecasting on the car class up to 3 seconds in the future. We evaluate forecasting performance on the pedestrian class in the appendix. As the test set is hidden, we follow [33] and conduct our analysis on the official validation split.

5.1. Metric Breakdown Analysis

In this section, we analyze different evaluation metrics by comparing a trivial constant position model to several state-of-the-art forecasting methods [33, 37, 43]. For the task of end-to-end forecasting from raw data, methods report both detection and forecasting confidence scores. For the simple constant position model, we threshold trajectories such that we only report those where the final position overlaps with the initial position (i.e the object is stationary) with high confidence. A good forecasting evaluation metric should indicate that the trivial constant position baseline is not a good predictor, as it only correctly predicts future locations of stationary objects and explicitly assumes a stationary world. Do existing metrics reveal this?

To answer this question, we report the results of our analysis in Table 1. We analyze the results using average and final displacement (ADE and FDE) errors at {60, 90}% recall [33], and a variant that averages results over all recall thresholds [49] (see Sec. 3.1 for a discussion of these metrics). Our trivial constant position baseline yields state-of-the-art results under the aforementioned evaluation settings. Current metrics are “gamable” by this trivial forecaster.

What about our forecasting mAP (Sec. 3.2)? We analyze these methods both through the lens of each motion class ($AP_{\text{stat.}}$, $AP_{\text{lin.}}$, and $AP_{\text{non-lin.}}$), and as an aggregate. First, we observe that the constant position model $AP_f$ evaluated over static cars performs better than FaF*, a state-of-the-art end-to-end forecaster. However, when we evaluate on the subset of cars that are in motion, $AP_{\text{lin.}}$ and $AP_{\text{non-lin.}}$ confirms that our metric behaves as expected: we obtain 0$mAP_f$ with the constant position model, indicating that it fails to predict the motion of moving cars. On the other hand, FaF* obtains 7.5 $AP_{\text{non-lin.}}$, indicating that motion forecasting from the raw sensory data is a very challenging problem. We observe that Trajectron++ outperforms the constant position model for moving objects (8.1 $AP_{\text{lin.}}$), but does not reach the performance of the constant position model or FaF* on stationary objects.

A good metric should summarize the performance on the full set of cars, i.e., in addition to predicting the motion of moving cars, a good model should correctly predict that parked cars are unlikely to move in the near future. This is achieved by our forecasting mAP that averages $AP_{\text{stat.}}$, $AP_{\text{lin.}}$ and $AP_{\text{non-lin.}}$. Our mAP$_f$ ranks state-of-the-art FaF* (31.5 mAP$_f$) favourably over the constant position baseline (22.1 mAP$_f$), as expected. Based on this analysis, we are confident we have the right tools to analyze FutureDet thoroughly!

5.2. Ablation and Comparison to State-of-the-Art

After confirming that our proposed forecasting mAP is a suitable metric for joint object detection and forecasting, we compare FutureDet to a number of baselines and two state-of-the-art methods.

Detection + Constant Velocity. We begin with an surprisingly simple, yet strong baseline which is often overlooked in forecasting literature. This baseline takes the detections, as well as the estimated velocity from our CenterPoint detector [53], and simply extrapolated forecasts as if objects are moving with constant velocity. Since CenterPoint is such a strong detector, this baseline produces strong results. Most of the ground-truth objects are approximately moving with a constant velocity, either moving directly forward or are stationary. We expect this model to under-perform on non-linear trajectories.

Detection + Forecast (FaF*, cf. [37]). This variant predicts a different velocity offset at every time step into the future for each detection, and derives trajectories by integrating velocities in the forward direction. This is precisely what Fast and Furious (FaF) does [37]. For an apples-to-apples comparison, we re-implement FaF using a CenterPoint-backbone and denote this model as FaF*. This method predicts a single trajectory per detection.

Trajectron++. We compare all of the aforementioned variants and ablations of our method to the state-of-the-art auto-
regressive trajectory prediction model, Trajectron++ [43]. This model is indicative of current state-of-the-art approaches for the traditional forecasting task where ground truth tracks are given. With this comparison, we wish to outline how the standard three-stage detection-tracking-forecasting approach compares with our end-to-end forecasting method. To construct this baseline, we begin with off-the-shelf detection and tracking results from CenterPoint [53]. CenterPoint performs tracking using the velocity offset estimates to match detections in each frame using a greedy matching between the current frame detections and previous frame detections. Trajectron++ then takes these predicted trajectories as input for forecasting.

**FutureDet.** We detect objects directly in future frames and backcast these future detections to the reference frame. Intuitively, the advantage of this variant over simple forecasting (FaF) is in that it encourages the network to learn a better feature representation for forecasting by placing "multiple bets" on the future position of objects in the current frame. As shown in Figure 2, this method naturally allows for a multiple-future interpretation of the observed sensory data (as discussed in Sec. 4). In Figure 3, we show qualitatively that our method can represent multiple futures. We note that the highest confidence future trajectories looks like constant velocity predictions (row 1), suggesting that this is indeed a challenging problem and constant velocity is a strong baseline. **FutureDet** consistently outperforms other baselines on non-linear trajectories. Notably, for $K = 5$, we improve the non-linear object forecasting accuracy by $4\%$ over FaF*. **FutureDet** trained with a PointPillars backbone provides modest improvement across metrics, and performs best overall.

**Discussion.** We compare the results of the aforementioned variants to **FutureDet** as well as Trajectron++ [43] in Table 2. First, we notice that moving object forecasting under our end-to-end setting is a challenging problem — none of the methods we study have high $AP_F$, suggesting the need for the community to focus on this problem. Second, despite the constant velocity model being conceptually simple, it performs on par with our FaF re-implementation and improves on Trajectron++ by +7.8 $mAP_F$. Unfortunately, this constant velocity baseline is usually under-emphasized in the literature. We argue here that it still serves as an important baseline. The poor performance of

---

**Table 1. Metric Breakdown Analysis:** We compare our simple constant position model to state-of-the-art prediction models, highlighting differences among various proposed metrics. ADE/FDE based metrics measured at different recall rates favor our trivial constant position baseline over state-of-the-art methods [33, 37, 43]. Only our proposed forecasting $mAP$ ($mAP_F$) favors state-of-the-art models over the constant position baseline. We report numbers for PnPNet [33] and SPF2 [49] from their respective papers. **Note:** Lower ADE/FDE is better and higher $AP$ is better.

| K=1 | $A_{det}$ | $A_P$ | $A_{det}$ | $A_P$ | $A_{det}$ | $A_P$ | $mAP$ | $A_{det}$ | $A_P$ | $A_{det}$ | $A_P$ | $mAP$ |
|-----|-----------|-------|-----------|-------|-----------|-------|-------|-----------|-------|-----------|-------|-------|
| Detection + Constant Velocity | 70.3 | 66.0 | 65.8 | 21.2 | 90.0 | 6.5 | 75.4 | 31.2 |
| Detection + Forecast (cf. [37]) | 69.1 | 64.7 | 66.1 | 22.2 | 86.3 | 7.5 | 73.8 | 31.5 |
| Trajectron++ [43] | 70.3 | 59.2 | 65.8 | 8.1 | 90.0 | 2.8 | 75.4 | 23.4 |

| K=5 | $A_{det}$ | $A_P$ | $A_{det}$ | $A_P$ | $A_{det}$ | $A_P$ | $mAP$ | $A_{det}$ | $A_P$ | $A_{det}$ | $A_P$ | $mAP$ |
|-----|-----------|-------|-----------|-------|-----------|-------|-------|-----------|-------|-----------|-------|-------|
| Detection + Constant Velocity | 70.3 | 66.0 | 65.8 | 21.2 | 90.0 | 6.5 | 75.4 | 31.2 |
| Detection + Forecast (cf. [37]) | 69.1 | 64.7 | 66.1 | 22.2 | 86.3 | 7.5 | 73.8 | 31.5 |
| Trajectron++ [43] | 70.3 | 59.2 | 65.8 | 8.1 | 90.0 | 2.8 | 75.4 | 23.4 |

Table 2. Joint car detection and forecasting evaluation on nuScenes. We adopt top-$K$ evaluation for forecasting and evaluate under two settings of $K = 1$ and $K = 5$ (for forecasting only). We further breakdown the performance of each model by examining the detection $AP$ ($AP_{det}$) and forecasting $AP$ ($AP_F$) on static, linear, and non-linearly moving sub-categories. First, we find that methods that are trained to detect objects in the current frame have higher overall $AP_{det}$ (Detection + Constant Velocity, row 1), while methods that are trained to detect objects in future frames have higher overall $AP_F$ (c.f. FutureDet, row 4), which is expected by design. For forecasting, surprisingly, Trajectron++ (row 3) is outperformed by constant velocity predictions (row 1), suggesting that this is indeed a challenging problem and constant velocity is a strong baseline. **FutureDet** consistently outperforms other baselines on non-linear trajectories. Notably, for $K = 5$, we improve the non-linear object forecasting accuracy by $4\%$ over FaF*. **FutureDet** trained with a PointPillars backbone provides modest improvement across metrics, and performs best overall.
Figure 3. We qualitatively evaluate forecasts from FutureDet. We denote ground-truth trajectories with green and multiple future predictions with blue for the highest confidence forecast and cyan for the remaining multiple-future predictions. Since we repurpose CenterPoint, a state-of-the-art detector, current frame detection performs well. Often, our model predicts that moving objects may be moving with constant velocity with high confidence. Given the data bias, where most objects are either stationary or are moving with constant velocity, this is a reasonable output. We highlight the multiple-future detection output in the top left.

Trajectron++ might also hint that performing direct end-to-end forecasting is advantageous over a three-stage approach of detection-tracking-forecasting, where errors can easily compound. FutureDet takes a different approach compared to existing methods. Our method improves upon all other baselines in terms of non-linear object $AP_f$ and the motion category-averaged $mAP_f$ (our primary metric). In addition, this multi-future interpretation also allows the performance to be improved in the $K = 5$ evaluation, where the forecast with minimum FDE from the top 5 ranked forecasts for each detection is evaluated. Note that for FutureDet $AP_f^{\text{static}}$, $K = 1$ results slightly decrease because bundling multiple trajectory estimates into one multi-future prediction for a single object reduces recall. However this is more than made up for in the increase in performance for detection and forecasting moving objects at $K = 5$. We train a version of our model with road masks as an additional input channel into the BEV feature representation (after the sparse-voxel backbone). This brings very little change to the results. We hypothesize that adding the map information does not provide additional information as it can be easily be learnt from the raw LiDAR input. However, further exploration is required to evaluate how to best fuse map information.

6. Conclusion

This paper presents a new end-to-end method for trajectory forecasting directly from LiDAR sensor data. Our proposed FutureDet is a natural forecasting-by-detection framework that allows for a multi-future interpretation of the observed evidence and establishes a new state-of-the-art. We provide thorough analysis of existing evaluation metrics for end-to-end forecasting and reveal that they can be gamed by a simple constant position model. To this end, we propose a new set of evaluation metrics based on the average precision metric that comprehensively evaluates joint detection and forecasting performance. This allows us to conduct a thorough analysis that reveals that a constant velocity model is a surprisingly strong baseline that should be considered in future forecasting work.

Limitations. As we do not explicitly enforce diverse trajectory generation, many of our multiple-futures are closely clustered. While FutureDet presents the first method for end-to-end forecasting from raw sensory data, capable of multi-future predictions, generation of diverse, multi-modal predictions remains an open challenge.

Acknowledgments. This work was supported by the CMU Argo AI Center for Autonomous Vehicle Research.
References

[1] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *IEEE Conf. Comput. Vis. Pattern Recogn.*, June 2016. 1, 3

[2] Javad Amirian, Jean-Bernard Hayet, and Julien Pettré. Social ways: Learning multi-modal distributions of pedestrian trajectories with gans. In *Conference on Computer Vision and Pattern Recognition Workshop*, 2019. 1, 3, 5

[3] Mehmet Aygün, Aljoša Ošep, Mark Weber, Maxim Maximov, Cyrill Stachniss, Jens Behley, and Laura Leal-Taixé. 4d panoptic lidar segmentation. In *IEEE Conf. Comput. Vis. Pattern Recogn.*, 2021. 2

[4] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, J. Gall, and C. Stachniss. Towards 3D LiDAR-based semantic understanding of 3D point cloud sequences: The SemanticKITTI Dataset. *Int. J. Robot. Res.*, 40(8-9):959–967, 2021. 2

[5] Apratim Bhattacharyya, Bernt Schiele, and Mario Fritz. Accurate and diverse sampling of sequences based on a “best of many” sample objective. In *IEEE Conf. Comput. Vis. Pattern Recogn.*, 2018. 1, 5

[6] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishna, Yu Pan, Giangarlo Baldan, and Oscar Beijbom. nuScenes: A multi-modal dataset for autonomous driving. In *IEEE Conf. Comput. Vis. Pattern Recogn.*, 2020. 1, 2, 3, 4, 6

[7] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giangarlo Baldan, and Oscar Beijbom. nuscenes: A multi-modal dataset for autonomous driving. In *IEEE Conf. Comput. Vis. Pattern Recogn.*, 2020. 2

[8] Junru Gu, Chen Sun, and Hang Zhao. DenseTNT: End-to-end trajectory prediction from dense goal sets. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15303–15312, 2021. 3

[9] Tomasz Kostrikov, Lin Chen, and Hang Zhao. Social-GAN: Socially acceptable trajectory generation with generative adversarial networks. In *IEEE Conf. Comput. Vis. Pattern Recogn.*, 2018. 1, 3, 5

[10] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, J. Gall, and C. Stachniss. Towards 3D LiDAR-based semantic understanding of 3D point cloud sequences: The SemanticKITTI Dataset. *Int. J. Robot. Res.*, 40(8-9):959–967, 2021. 2

[11] Patrick Dendorfer, Sven Elfllein, and Laura Leal-Taixé. Mogan: A multi-generator model preventing out-of-distribution samples in pedestrian trajectory prediction. In *Int. Conf. Comput. Vis.*, 2021. 1, 3, 5

[12] Patrick Dendorfer, Aljoša Ošep, and Laura Leal-Taixé. Goalgan: Multimodal trajectory prediction based on goal position estimation. In *Asian Conf. Comput. Vis.*, 2020. 1, 3, 5

[13] M. Everingham, L. Van Gool, C.K.I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (VOC) challenge. *Int. J. Comput. Vis.*, 88(2):303–338, 2010. 3, 4

[14] Davi Frossard and Raquel Urtasun. End-to-end learning of multi-sensor 3D tracking by detection. In *IEEE Int. Conf. Robotics and Autom.*, 2018. 2

[15] Jiayang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. Vectornet: Encoding hd maps and agent dynamics from vectorized representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11525–11533, 2020. 3

[16] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *IEEE Conf. Comput. Vis. Pattern Recogn.*, 2012. 2

[17] Thomas Gilles, Stefano Sabatini, Dzmitry Tsishkou, Bogdan Stanciulescu, and Fabien Moutarde. Home: Heatmap output for future motion estimation. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC),* pages 500–507. IEEE, 2021. 3

[18] Jiyang Gao, Chen Sun, Hang Zhao. Social-GAN: Socially acceptable trajectory generation with generative adversarial networks. In *IEEE Conf. Comput. Vis. Pattern Recogn.*, 2018. 1, 3, 5

[19] Dirk Helbing and Péter Molnár. Social force model for pedestrian dynamics. *Physical Review E*, 1995. 2

[20] Boris Ivanovic and Marco Pavone. The trajectory: Probabilistic multi-agent trajectory modeling with dynamic spatiotemporal graphs. In *Int. Conf. Comput. Vis.*, 2019. 1, 3, 5

[21] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014. 3

[22] Vineet Kosaraju, Aim Sadeghian, Roberto Martin-Martín, Ian Reid, Hamid Rezatofighi, and Silvio Savarese. Social-GAT: Multimodal trajectory forecasting using bicycle-gan and graph attention networks. In *Neural Information Processing Systems*, 2019. 1, 3, 5

[23] Mohamed Kosaraju, Aim Sadeghian, Roberto Martin-Martín, Ian Reid, Hamid Rezatofighi, and Silvio Savarese. Social-GAT: Multimodal trajectory forecasting using bicycle-gan and graph attention networks. In *Neural Information Processing Systems*, 2019. 1, 3, 5

[24] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet classification with deep convolutional neural networks. In *Adv. Neural Inform. Process. Syst.*, 2012. 2

[25] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *IEEE Conf. Comput. Vis. Pattern Recogn.*, 2019. 2

[26] Laura Leal-Taixé, Gerard Pons-Moll, and Bodo Rosenhahn. Everybody needs somebody: Modeling social and grouping behavior on a linear programming multiple people tracker. In *ICCV Workshops*, 2011. 2
