A. Appendix

A.1. Object Detection Losses

Focal Loss. Focal Loss [42] is designed to address class imbalance by down-weighting well-classified examples and focusing on harder, misclassified examples. The Focal Loss function is defined as:

$$\mathcal{L}_{\text{FL}}(p) = -(1-p)^{\gamma}\log(p),$$

where p is the model's estimated probability for the correct class, and γ is a tunable parameter that controls the focus on harder examples. When $\gamma = 0$, Focal Loss reduces to standard cross-entropy loss. As γ increases, the loss function places greater emphasis on difficult examples, which is useful in contexts like object detection where class imbalance is common. This is particularly effective when most examples are easy negatives (e.g., background) and would otherwise dominate the loss.

Gaussian Focal Loss. Gaussian Focal Loss (GFL) [38, 70] is a variant of Focal Loss specifically designed for 3D object detection. In this approach, object centers are represented using a Gaussian heatmap, and the loss function focuses on the points near the center of the object. The Gaussian Focal Loss function is given by:

$$\mathcal{L}_{\text{GFL}} = -(1-\hat{y})^{\eta}\log(p),$$

where $\hat{y}$ is the Gaussian-distributed ground truth centered around the object, and p is the predicted probability. The term $(1-\hat{y})^{\eta}$ down-weights points far from the object's center, ensuring that the model focuses on improving localization precision for points near the center. The parameter η controls the degree to which points further from the center are down-weighted.

A.2. Derivation of the Loss

Loss function. We aim to derive the loss function for multi-label classification using the Beta distribution by computing the Bayes risk with respect to the class predictor. The probability of class j for instance i is modeled as a Beta distribution, $\text{Beta}(\boldsymbol{\alpha}_{ij}, \boldsymbol{\beta}_{ij})$. The resulting loss is given by:

$$\mathcal{L}_i(\Theta) := \int \left[\sum_{j=1}^{C} -y_{ij}\log(p_{ij})\right] \frac{1}{B(\boldsymbol{\alpha}_{ij}, \boldsymbol{\beta}_{ij})} \prod_{j=1}^{C} p_{ij}^{\boldsymbol{\alpha}_{ij}-1}(1-p_{ij})^{\boldsymbol{\beta}_{ij}-1}\, \mathrm{d}\mathbf{p}_i, \tag{4}$$

where $B(\boldsymbol{\alpha}_{ij}, \boldsymbol{\beta}_{ij})$ is the Beta function, defined as:

$$B(\boldsymbol{\alpha}_{ij}, \boldsymbol{\beta}_{ij}) = \frac{\Gamma(\boldsymbol{\alpha}_{ij})\Gamma(\boldsymbol{\beta}_{ij})}{\Gamma(\boldsymbol{\alpha}_{ij}+\boldsymbol{\beta}_{ij})}, \tag{5}$$

and $\Gamma(\cdot)$ is the Gamma function. For each class j, we consider the individual term:

$$\mathcal{L}_i(\Theta) = \sum_{j=1}^{C} \int -y_{ij}\log(p_{ij}) \frac{p_{ij}^{\alpha_{ij}-1}(1-p_{ij})^{\beta_{ij}-1}}{B(\boldsymbol{\alpha}_{ij}, \boldsymbol{\beta}_{ij})}\, \mathrm{d}p_{ij}. \tag{6}$$

The integral term is the expected value of $-\log(p_{ij})$ with respect to a Beta distribution. The expectation of $\log(p_{ij})$ under a Beta distribution $\text{Beta}(\boldsymbol{\alpha}_{ij}, \boldsymbol{\beta}_{ij})$ is given by:

$$\mathbb{E}_{p_{ij}\sim\text{Beta}(\boldsymbol{\alpha}_{ij},\boldsymbol{\beta}_{ij})}\left[\log(p_{ij})\right] = \psi(\boldsymbol{\alpha}_{ij}) - \psi(\boldsymbol{\alpha}_{ij}+\boldsymbol{\beta}_{ij}), \tag{7}$$

.

Thus, applying the expectation for both p_{ij} and $(1-p_{ij})$, the loss function becomes:

$$\mathcal{L}_i(\Theta) = \sum_{j=1}^{C} \left[y_{ij}\left(\psi(\boldsymbol{\alpha}_{ij}+\boldsymbol{\beta}_{ij}) - \psi(\boldsymbol{\alpha}_{ij})\right) + (1-y_{ij})\left(\psi(\boldsymbol{\alpha}_{ij}+\boldsymbol{\beta}_{ij}) - \psi(\boldsymbol{\beta}_{ij})\right)\right]. \tag{8}$$

A.3. Derivation of the Regularization Term

We want to compute the Kullback-Leibler (KL) divergence between the predicted Beta distribution $\text{Beta}(\tilde{\boldsymbol{\alpha}}_i, \tilde{\boldsymbol{\beta}}_i)$ and the prior Beta distribution $\text{Beta}(\mathbf{1}, \mathbf{1})$. The KL divergence between two distributions $P(x)$ and $Q(x)$ is given by:

$$\text{KL}(P\|Q) = \int P(x) \log \frac{P(x)}{Q(x)} \, \mathrm{d}x.$$

The Beta distribution is parameterized by two values, α and β, and is given by:

$$\text{Beta}(x \mid \alpha, \beta) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{B(\alpha, \beta)},$$

where $B(\alpha, \beta)$ is the Beta function that normalizes the distribution and is defined as:

$$B(\alpha, \beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)},$$

with $\Gamma(\cdot)$ being the Gamma function. For two Beta distributions $\text{Beta}(\alpha_1, \beta_1)$ and $\text{Beta}(\alpha_2, \beta_2)$, the KL divergence is computed as:

$$\text{KL}\left(\text{Beta}(\alpha_1, \beta_1)\|\text{Beta}(\alpha_2, \beta_2)\right) = \int_0^1 \text{Beta}(x \mid \alpha_1, \beta_1) \log \frac{\text{Beta}(x \mid \alpha_1, \beta_1)}{\text{Beta}(x \mid \alpha_2, \beta_2)} \, \mathrm{d}x.$$

Substituting the expressions for both Beta distributions, we get:

$$\frac{\text{Beta}(x \mid \alpha_1, \beta_1)}{\text{Beta}(x \mid \alpha_2, \beta_2)} = \frac{x^{\alpha_1-1}(1-x)^{\beta_1-1}B(\alpha_2, \beta_2)}{x^{\alpha_2-1}(1-x)^{\beta_2-1}B(\alpha_1, \beta_1)}.$$

Simplifying, we obtain:

$$\frac{\text{Beta}(x \mid \alpha_1, \beta_1)}{\text{Beta}(x \mid \alpha_2, \beta_2)} = \frac{B(\alpha_2, \beta_2)}{B(\alpha_1, \beta_1)} x^{\alpha_1-\alpha_2}(1-x)^{\beta_1-\beta_2}.$$

Thus, the KL divergence becomes:

$$\begin{aligned}\text{KL}(\text{Beta}(\alpha_1, \beta_1)\|\text{Beta}(\alpha_2, \beta_2)) = \; & \log \frac{B(\alpha_2, \beta_2)}{B(\alpha_1, \beta_1)} \\ & + (\alpha_1 - \alpha_2) \int_0^1 \text{Beta}(x \mid \alpha_1, \beta_1) \log x \; \mathrm{d}x \\ & + (\beta_1 - \beta_2) \int_0^1 \text{Beta}(x \mid \alpha_1, \beta_1) \log(1-x) \; \mathrm{d}x.\end{aligned}$$

From known properties of the Beta distribution, we have the following results:

$$\int_0^1 \text{Beta}(x \mid \alpha_1, \beta_1) \log x \; \mathrm{d}x = \psi(\alpha_1) - \psi(\alpha_1 + \beta_1),$$

and

$$\int_0^1 \text{Beta}(x \mid \alpha_1, \beta_1) \log(1-x) \; \mathrm{d}x = \psi(\beta_1) - \psi(\alpha_1 + \beta_1),$$

where $\psi(\cdot)$ is the digamma function, defined as the derivative of the logarithm of the Gamma function, $\psi(x) = \frac{\mathrm{d}}{\mathrm{d}x} \log \Gamma(x)$. Substituting these results into the KL divergence formula, we get the full expression:

$$\text{KL}\left(\text{Beta}(\alpha_1, \beta_1)\|\text{Beta}(\alpha_2, \beta_2)\right) = \log \frac{B(\alpha_2, \beta_2)}{B(\alpha_1, \beta_1)} + (\alpha_1 - \alpha_2)(\psi(\alpha_1) - \psi(\alpha_1 + \beta_1)) + (\beta_1 - \beta_2)(\psi(\beta_1) - \psi(\alpha_1 + \beta_1)).$$

A.4. Experimental Details and Ablations

In this subsection, we provide additional details of our experiments from the main body of the paper, such as the parameters of the uncertainty baselines, training details, and the choice of threshold. We also include several ablations, demonstrating that the quality and inference speed of the detector remain unchanged, and that the results are stable across different random seeds used for training.

A.4.1. 3D Object Detector Accuracy

As mentioned in the main body of the paper, our uncertainty method does not compromise the original detection quality. Since we fine-tune only a small subset of the model parameters, the original detector's performance is preserved. As a result, the detection quality (measured by IoU) remains virtually unchanged, as shown in Tab. 4: 70.5 for FocalFormer (C), 66.4 for FocalFormer (C+L), and 65.5 for DeformFormer (L), closely matching the original results reported in[11]. This demonstrates that our uncertainty estimation mechanism introduces no degradation to the base model's capabilities.

Table 4. **Detection Quality Comparison Before and After Applying Evidential Learning.** Our method preserves the original mAP and NDS performance.

Model	Original Model		Our Model	
	mAP (%)	NDS (%)	mAP (%)	NDS (%)
FocalFormer (C+L)	70.5	73.1	70.5	73.1
FocalFormer (L)	66.4	70.9	66.4	70.9
DeformFormer (L)	65.5	70.7	65.5	70.7

A.4.2. Inference Speed

We conduct a latency analysis of FocalFormer3D on the nuScenes dataset. All runtimes are measured on the same V100 GPU for fair comparison. As shown in Tab. 5, the computation time is dominated by the convolution-based backbone, while our modified BEV-level evidential learning head introduces only a negligible increase in latency.

Table 5. **Inference Latency.** We report the latency of FocalFormer3D before and after applying evidential learning, measured on the same V100 GPU. The results show that our evidential learning head introduces a negligible overhead (1 ms).

Model	Original Model Latency (ms)	Our Model Latency (ms)
FocalFormer3D (Lidar)	109	110

A.4.3. Choice of Thresholds

As for the IoU threshold used in Sec. 4.3, we chose a value of 0.3 as it is used in the 3D object detection literature [66] to conservatively determine whether an object is correctly detected. In this context, an IoU below 0.3 typically indicates a poorly localized or missed detection, making it a meaningful threshold.

For the distance thresholds $d = 2$ and $d = 4$ used in the missed detection experiments in Sec. 4.4, these values are drawn from the standard evaluation distances used in the NuScenes benchmark [11] (Section 4.3, Figure 5), which includes $d = 0.5, 1.0, 2.0$, and 4.0 meters. We selected $d = 2$ and $d = 4$ to focus on mid-to-large object ranges, aiming for a robust evaluation of missed detections that are practically relevant in autonomous driving scenarios.

A.4.4. Parameters of Uncertainty Baselines

Across all methods, we use 5 samples, as recommended in the original papers and in [14]—this includes 5 ensemble members for Deep Ensembles, 5 stochastic forward passes for MC-Dropout, and 5 submodels for Masksembles, BatchEnsemble, and PackedEnsemble. For MC-Dropout, we additionally follow [14] by applying a dropout rate of 0.1 to the last two layers of the model. For Masksembles, we use a scaling factor of 2.0, as proposed in the original paper. For Deep Ensembles [35], we rely on the standard PyTorch [55] initialization, a common and well-established choice in the literature. BatchEnsemble [65] only requires setting the number of submodels (that we fix to 5), and for PackedEnsemble, we adopt the original configuration of the PackedEnsemble layer as described in [37].

All methods are applied on top of a pre-trained 3D detector, trained following the protocol in [11]. We then replace the BEV-level heatmap head with an uncertainty-aware counterpart. For all methods except Deep Ensembles, this requires only a single fine-tuning stage, using Focal Loss for 3 epochs with the Adam [34] optimizer. For Deep Ensembles, we fine-tune the head independently for each ensemble member.

A.4.5. Multiple Random Seeds

To demonstrate the robustness and statistical significance of our improvements, we report the mean and variance of key metrics across three independent runs using different random seeds. Tab. 6 and Tab. 7 present results for two LiDAR-based architectures: FocalFormer and DeformFormer, each trained with 10k initially labeled scenes. These results confirm that the observed gains are consistent and not due to random variation.

Table 6. **Detection Performance for FocalFormer (LiDAR-based).** Results are averaged over 3 runs with different random seeds.

Metric	Pseudo	Ours
IoU	59.38 ± 0.09	59.59 ± 0.04
NDS	55.64 ± 0.04	55.83 ± 0.06

Table 7. **Detection Performance for DeformFormer (LiDAR-based).** Results are averaged over 3 runs with different random seeds.

Metric	Pseudo	Ours
IoU	60.88 ± 0.05	61.09 ± 0.06
NDS	57.25 ± 0.07	57.66 ± 0.05

These results highlight the consistency of our approach across architectures and confirm the statistical significance of the improvements. The standard deviations remain low, suggesting stable training behavior even in low-data regimes.

Finally, on the three intermediary tasks—OOD detection, bounding box error identification, and missed object detection—our method consistently outperforms baselines by margins often exceeding 10%. While the per-run variability is within 1–3%, these improvements remain statistically meaningful. Together, these results reinforce the effectiveness and reliability of our uncertainty estimation framework in real-world deployment scenarios.

A.5. Uncertainty in Object Detection

In this subsection, we provide a more extensive discussion of existing uncertainty methods applied to object detection and highlight their limitations compared to our approach. To the best of our knowledge, none of the existing methods simultaneously model all types of uncertainty, require minimal architectural changes, support pretrained models, preserve inference speed, and extend beyond box-level uncertainty (unlike ours, which operates at the BEV level). These limitations significantly reduce their applicability, whereas our proposed method satisfies all of these criteria.

In Tab. 8, we compare existing methods across several metrics that reflect the limitations and capabilities mentioned above. Specifically, the *"Aleatoric"* column indicates whether the method can model aleatoric uncertainty, while *"Epistemic"* shows whether epistemic uncertainty can be captured. *"Pretrained"* denotes whether the method can be applied to a pretrained model. *"Beyond Box Unc."* indicates whether the uncertainty estimates go beyond individual predicted boxes (i.e., beyond detection head outputs), which is essential for tasks like scene-level OOD detection or missed object detection. *"Speed"* reflects whether the method significantly slows down inference. Finally, *"LiDAR"* and *"Multi-View"* indicate the input modalities that the method supports / was applied to.

LiDAR 3D Uncertainty [19] applies a well-known uncertainty estimation approach [25] using multiple forward passes with MC-Dropout activated at the detection head (which predicts box parameters and class probabilities). Uncertainty is captured through the variability of these predictions, but it is limited to the detection head only—that is, it estimates uncertainty solely for individual predicted bounding boxes. As a result, this type of uncertainty is not applicable to tasks such as OOD scene detection (Sec. 4.2) or missed object detection (Sec. 4.4).

LiDAR 3D Uncertainty++[20] is similar to the previous approach, but instead of using MC-Dropout, it predicts uncertainty as an additional output variance[33], and thus can only model aleatoric uncertainty. While this addresses the issue of prohibitively expensive inference in the prior method, the predicted uncertainties are still limited to the level of individual bounding boxes, and the method cannot be applied to a pretrained object detector.

LaserNet [49] introduced a novel architecture for LiDAR-based object detection that produces probabilistic predictions for 3D box parameters by predicting the variance of box corners. This approach estimates aleatoric uncertainty only, operates solely at the level of individual bounding boxes, and cannot be applied to pretrained models.

LaserNet++ [48] builds on the original LaserNet by introducing a different method for producing probabilistic predictions over 3D box parameters, explicitly accounting for potential noise in the ground-truth annotations. While it improves upon the

Table 8. **Existing Uncertainty Methods for Object Detection.** Comparison of existing uncertainty estimation methods for 3D object detection across key criteria. The table summarizes each method's ability to model aleatoric and epistemic uncertainty, support pretrained models, extend beyond box-level uncertainty, preserve inference speed, and operate with LiDAR or multi-view inputs. Our method is the only one satisfying all criteria.

Method	*Aleatoric*	*Epistemic*	*Pretrained*	*Beyond Box Unc.*	*Speed*	*Lidar*	*Multi-View*
Lidar 3D Uncertainty [19]	✓	✓	✗	✗	✗	✓	✗
Lidar 3D Uncertainty++ [20]	✓	✗	✗	✗	✓	✓	✗
LaserNet [49]	✓	✗	✗	✗	✓	✓	✗
LaserNet++ [48]	✓	✗	✗	✗	✓	✓	✗
GUP [44]	✓	✗	✗	✗	✓	✗	✗
U-SPA [68]	✓	✓	✗	✗	✗	✓	✗
EvCenterNet [52]	✓	✓/✗	✗	✓	✓	✗	✗
TMNF [21]	✓	✓	✓	✗	✓	✗	✗
AB3DMOT [53]	✓	✗	✗	✗	✓	✓	✗
uPLAM [58]	✓	✓	✗	✓	✓	✗	✗
Our Method	✓	✓	✓	✓	✓	✓	✓

original method in terms of performance, LaserNet++ suffers from the same limitations: it only estimates aleatoric uncertainty, operates at the level of individual bounding boxes, and cannot be applied to pretrained models.

GUP [44] focuses on monocular 3D detection, which differs fundamentally from our setting. It addresses challenges specific to monocular depth ambiguity, which do not arise in LiDAR-based or multi-view detection. Its uncertainty formulation is tightly coupled with monocular constraints and does not transfer to multi-modal or LiDAR-only setups. Additionally, the method models only aleatoric uncertainty and cannot be applied on top of pretrained models.

U-SPA [68] estimates uncertainty at the level of individual predicted bounding boxes, focusing on objectness and geometric confidence via a Laplace approximation over detection head weights. Like previous approaches, it cannot be applied to pretrained models and introduces significant computational overhead due to its reliance on sampling.

EvCenterNet [52] applies evidential learning to 2D object detection, focusing mainly on aleatoric uncertainty to improve detection quality. In contrast, our method emphasizes epistemic uncertainty, which is crucial for handling out-of-distribution and rare cases in safety-critical 3D environments. EvCenterNet models uncertainty using a Dirichlet distribution over objectness heatmaps, while we use a Beta distribution tailored for BEV-based 3D detection. Moreover, EvCenterNet requires major architectural changes, making it incompatible with pretrained models, whereas our method is lightweight and easily integrates with existing detectors.

TMNF [21] addresses OOD detection at the bounding box level in 2D images. While relevant in spirit, it is limited to image-based 2D settings and does not generalize to 3D scenes or BEV-based reasoning—especially in configurations where image data is absent (e.g., LiDAR-only setups). Moreover, it shares the same limitations as previously discussed approaches, as it estimates uncertainty only for predicted boxes and cannot capture uncertainty in undetected or ambiguous regions.

AB3DMOT [53] estimates uncertainty at the level of individual predicted bounding boxes, primarily focusing on objectness and geometric confidence. However, this approach limits uncertainty estimation to existing detections, making it ineffective in capturing uncertainty in undetected, ambiguous, or out-of-distribution regions. In contrast, our method operates on BEV features and produces uncertainty estimates across the entire scene grid. This enables the identification of spatial regions with high uncertainty—even where no objects are detected—which is crucial for tasks like OOD detection and active auto-labeling.

uPLAM [58] also employs a Dirichlet-based evidential uncertainty model but focuses solely on semantic-level uncertainty estimation without directly addressing 3D object detection.

A.6. Risk Management in Motion Planning

Uncertainty estimation is a critical component in risk-aware motion planning, particularly in autonomous driving applications. While our primary objective is to demonstrate (i) the ability to obtain reliable uncertainty estimates and (ii) their application in the auto-labeling pipeline, our framework also has important implications for risk management.

To better understand its potential in this context, we conducted additional experiments assessing how uncertainty estimation

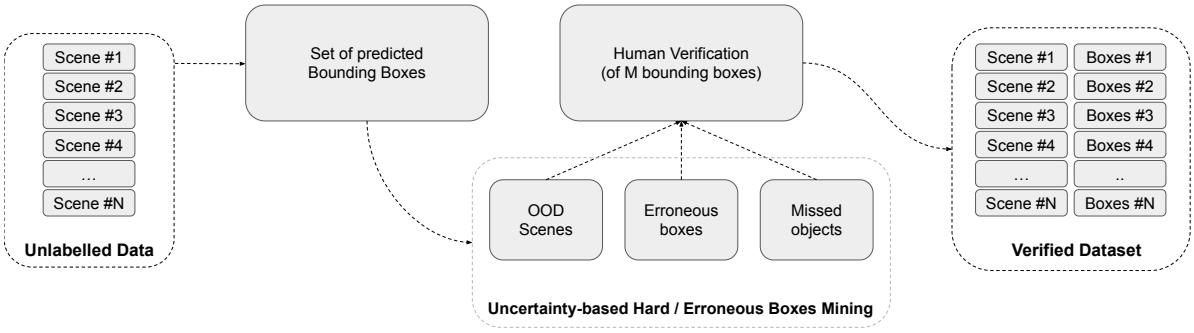

Figure 4. **Uncertainty-Guided Auto-Labeling Pipeline.** The pipeline improves pseudo-labeling in 3D object detection by integrating uncertainty estimation. A trained detector generates bounding box predictions for unlabelled scenes, which are refined through human verification. Uncertainty estimation determines which boxes require verification based on three categories: (i) scene-level uncertainty for out-of-distribution (OOD) instances, (ii) bounding box-level uncertainty for erroneous predictions, and (iii) missed object detection. This targeted refinement optimizes relabeling efforts, enhancing label quality and detection performance.

can mitigate safety risks. Our findings indicate that in nearly 24% of scenes, the 3D detector fails to recognize at least one object. When these perception outputs are fed into a downstream planner responsible for trajectory generation, undetected objects can lead to hazardous situations. In the worst case, the planner may inadvertently produce a trajectory that collides with an unseen obstacle. By incorporating uncertainty estimation into the perception pipeline, our method helps identify scenarios where the model is likely to miss objects, enabling fallback strategies such as driver intervention or adaptive planning adjustments. Without this uncertainty-aware mechanism, the system lacks a way to flag potentially dangerous errors. In contrast, our approach, similar to the method outlined in Section 3, allows for the detection of 10–20% of these high-risk cases. Specifically, it captures approximately 15% of such cases for the LiDAR-based FocalFormer model and 11% for the DeformFormer model, as summarized in Table 9. By reducing the likelihood of undetected objects influencing planning decisions, our method enhances safety and contributes to more robust trajectory generation in real-world driving scenarios.

Table 9. Recall (%) of high-risk situations detected using our uncertainty estimation framework.

	FocalFormer (L)	**DeformFormer (L)**
Recall, %	14.9	11.2

A.7. Uncertainty-Guided Auto-Labeling Pipeline

To enhance pseudo-labeling quality in 3D object detection, we incorporate uncertainty estimation into the auto-labeling process. Below, we outline the structure of our pipeline and how uncertainty is leveraged to improve label reliability.

Pipeline Overview Our method begins by training a 3D object detector on a randomly selected subset of NuScenes training scenes (10,000 or 20,000 scenes in our experiments). The trained model then generates pseudo-labels for the remaining unlabeled data. To refine these pseudo-labels, we apply our uncertainty estimation framework, which analyzes uncertainty at three levels: scene-level, where we flag entire scenes that contain high uncertainty and may indicate out-of-distribution (OOD) instances; bounding box-level, where we highlight individual predictions that require correction; and missed object detection, where we identify regions of high uncertainty that may contain undetected objects. These three levels of uncertainty guide our label refinement strategy, as illustrated in Figure 1 and discussed below.

Uncertainty-Guided Label Refinement To make efficient use of labeling resources, we allocate a fixed relabeling budget across three key areas:

- **Scene-Level Relabeling**: We prioritize scenes with the highest uncertainty, assuming that all objects in these scenes are re-annotated, consuming the scene-level relabeling budget. For example, if a scene contains 10 ground-truth objects, 10 boxes are deducted from the budget.

- **Bounding Box Verification**: After using the scene-level budget, we refine pseudo-labels by verifying and correcting individual bounding boxes with the highest uncertainty scores.
- **Missed Object Correction**: Any remaining budget is allocated to identifying and correcting false negatives in high-uncertainty regions.

This approach ensures that relabeling resources are distributed effectively, focusing on the most uncertain predictions. Training a model with these refined labels results in significant performance gains, demonstrating the effectiveness of uncertainty-guided pseudo-labeling.

Dataset and Budget Allocation Table 10 outlines the dataset configuration used in our experiments, while Table 11 details how uncertainty estimation guides the relabeling process.

Table 10. Dataset setup for different pseudo-labeling strategies.

Method	Initial Scenes	Unlabeled Pool	Description
Random Selection	Nk	$30 - N$k	Train detector on randomly selected scenes.
Pseudo-Labeling	Nk	$30 - N$k	Generate pseudo-labels and retrain on 30k scenes.
Ours (PL + Verification)	$N - 1$k	$31 - N$k	Correct pseudo-labels for 1k scenes (30k boxes).

Table 11. Allocation of relabeling budgets based on uncertainty estimation.

Uncertainty Use Case	Budget (Boxes)	Description
Scene-Level Uncertainty	10,000	Relabel all objects in the most uncertain scenes.
Bounding Box-Level Uncertainty	10,000	Verify and correct high-uncertainty bounding boxes.
Missed Object Detection	10,000	Identify and recover undetected objects in uncertain regions.

A.8. Reduction in Re-Annotation Volume

To demonstrate the efficiency of our method in reducing re-annotation volume compared to existing active learning approaches, we provide quantitative evidence in Tables 12 and 13. These results show that, for a given target IoU, our approach consistently requires fewer labeled samples while achieving higher NDS scores than both Random and Pseudo-Labeling baselines.

By leveraging uncertainty-guided relabeling, our method optimizes annotation costs while maintaining high detection performance. As shown in Table 12, our approach reduces the number of labeled samples by 2,200 compared to the Random baseline, leading to a 12.5% cost reduction (assuming a constant dollar cost to label each scene)and a significant improvement over the Pseudo Labeling baseline without uncertainty. This improvement increases the scalability of the annotation process by maximizing the number of labeled instances within a fixed budget. Similarly, Table 13 highlights a reduction of 4,400 labeled samples, corresponding to a 30% cost savings.

These results span different sensor configurations, including LiDAR-only and Camera + LiDAR setups, demonstrating the generalizability of our framework. Across all configurations, our approach consistently outperforms baselines while requiring fewer re-annotated samples, confirming its effectiveness in optimizing labeling efficiency.

Table 12. Labeled samples and NDS for IoU = 0.62 (FocalFormer, LiDAR). Our method reduces annotation requirements while improving detection performance. Ps.-Lab. = Pseudo-Labeling.

Base Model	Random	Ps.-Lab.	Ours
# Samples	17,600	16,800	15,400
NDS	0.5651	0.5699	0.5817

Table 13. Labeled samples and NDS for IoU = 0.68 (FocalFormer, Camera + LiDAR). Our method achieves better NDS with fewer labels.

Base Model	Random	Ps.-Lab.	Ours
# Samples	14,500	11,600	10,100
NDS	0.5965	0.5979	0.6004