AUTOPRUNE: ACCELERATING END-TO-END DRIVING TRAINING VIA GRADIENT-PRESERVING ADAPTIVE DATA PRUNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Unified end-to-end architectures have reshaped autonomous driving, but their reliance on massive datasets leads to weeks-long training, severely hindering model iteration. A natural way to accelerate training is to remove simple or redundant data that provide limited learning value. However, existing data reduction methods for end-to-end driving often rely on static data features and hand-crafted selection rules, facing two fundamental limitations: (1) Gradient distortion. Pruning low-value data can alter the gradient composition and degrade model performance; (2) Training-dynamics misalignment. Evolving data values make fixed pruning decisions stale, while fewer iterations can misalign the optimization schedule with full-data progress. Therefore, we present AutoPrune, which reduces training-data usage to accelerate end-to-end driving training while mitigating pruning-induced performance degradation. First, Gradient-Preserving Data Pruning is introduced to compensate for the missing training gradients caused by pruning low-loss samples, so that the corrected aggregate gradient matches its full-data counterpart in expectation. Second, Training-Dynamics Adaptation uses loss-adaptive sample gating to update pruning decisions from the latest available loss records and pruning-aware learning-rate alignment to follow the corresponding full-data training progress. Experiments across multiple end-to-end driving models show that AutoPrune retains performance close to the corresponding full-data references at a 40% overall pruning ratio. Comparative and ablation studies further support the effectiveness of the proposed components.

1 INTRODUCTION

Unified end-to-end architectures have substantially advanced autonomous driving by jointly optimizing perception, prediction, and planning. However, their growing reliance on large-scale driving datasets has made training increasingly expensive: at industry scale, a single model iteration can consume tens of thousands of GPU-hours (Liu et al., 2025a). Meanwhile, further data scaling brings rapidly diminishing returns; achieving an additional 5% reduction in FDE is estimated to require more than 33 times the original amount of training data (Naumann et al., 2025). Such inefficiency is further exacerbated by the highly repetitive composition of driving datasets; for example, 96% of the data aggregated from Waymo, nuPlan, Lyft and INTERACTION are non-interactive (Jiang et al., 2025b). Together, these observations suggest that a substantial portion of the training cost arises from repeatedly processing large volumes of data that provide limited marginal value, motivating pruning redundant or low-value samples to accelerate training.

Existing data-pruning methods can be broadly categorized into static and dynamic paradigms. Static methods construct a fixed subset before training, typically according to scene attributes, representation coverage, or distributional diversity (Sadat et al., 2021; Greer & Trivedi, 2025). Such methods implicitly assume that the value of each data sample remains stable throughout optimization. In practice, however, data value is model-dependent and evolves with the learning state: samples that are informative at early stages may become redundant once they have been sufficiently learned, whereas samples initially considered unimportant may become useful as the model evolves. A fixed subset therefore becomes progressively misaligned with the current learning state, leading to value staleness. Dynamic methods mitigate this limitation by using model feedback, such as loss, predic-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

tive uncertainty, or intermediate representations, to periodically reassess data value and update the data participating in training (Segal et al., 2022; Luo et al., 2023a; Lei et al., 2025).

However, data pruning is not merely about deciding which data to retain. Removing data also alters the optimization signal and changes the progression of training, revealing two fundamental challenges. *(1) Gradient distortion.* Data considered low-value still contribute non-zero gradients during optimization. Directly removing them changes the composition of gradients used for optimization and may cause the resulting gradient distribution to deviate from that of full-data training. Such deviations may alter parameter updates and affect final performance. *(2) Training-dynamics misalignment.* Effective pruning must track both the evolving value of data and the corresponding optimization progress. Though existing dynamic methods periodically update the data participating in training according to the current model state, they typically retain the optimization schedule designed for full-data training. Consequently, data selection evolves with the model, while the optimization schedule may no longer correspond to the same stage of full-data training.

To this end, we propose AutoPrune, a general data-pruning framework for accelerating end-to-end driving training. AutoPrune addresses gradient distortion and training-dynamics misalignment through two complementary components. *(1) Gradient-Preserving Data Pruning.* AutoPrune identifies low-loss samples as pruning candidates and randomly removes a subset of them. It applies inverse-probability correction to the retained candidate samples to compensate the gradient-distribution shift introduced by pruning. The corrected aggregate gradient is theoretically shown to match the corresponding full-data aggregate gradient in expectation over the random pruning operation. *(2) Training-Dynamics Adaptation.* AutoPrune further adapts the pruning process to the evolving training state. Loss-adaptive sample gating periodically refreshes pruning decisions using the latest available loss records, preventing a fixed selection criterion from becoming stale as training progresses. Meanwhile, pruning-aware learning-rate alignment adjusts the learning-rate schedule to match the corresponding full-data training progress. Together, these mechanisms adapt data selection and schedule progression to the evolving training process. The main contributions of this paper are as follows:

- We introduce AutoPrune, a training acceleration framework for end-to-end driving that reduces redundant training data while achieving performance comparable to full-data training.
- Gradient-Preserving Data Pruning is developed to mitigate pruning-induced gradient distribution shift through gradient compensation, while preserving the corresponding full-data aggregate gradient in expectation.
- Training-Dynamics Adaptation is introduced to update pruning decisions via loss-adaptive sample gating and align the optimization schedule under pruning with that of full-data training via pruning-aware learning-rate alignment, allowing data pruning to adapt to the evolving training process.
- Experiments on multiple end-to-end driving models demonstrate that AutoPrune achieves full-data performance even when 40% of the training data are pruned. Comparative and ablation experiments further validate its superiority and the benefits of each component.

2 Related Work

Existing approaches to reducing driving data can be broadly divided into static and dynamic selection, depending on whether data value is estimated before training or updated using feedback from the evolving model.

2.1 Static Pruning Based on Data Characteristics

Static methods estimate data value before training and keep the selected subset fixed throughout optimization. Early efforts used manually designed scene descriptors, such as traffic participants, road structures, and scene complexity, to favor representative and diverse driving data (Sadat et al., 2021). Yang & Su introduced a more structured objective, formulating selection as a coreset problem that removes redundancy while balancing data difficulty and enables driving models to approach full-data performance using only an Argoverse subset. More recent studies use richer, task-related

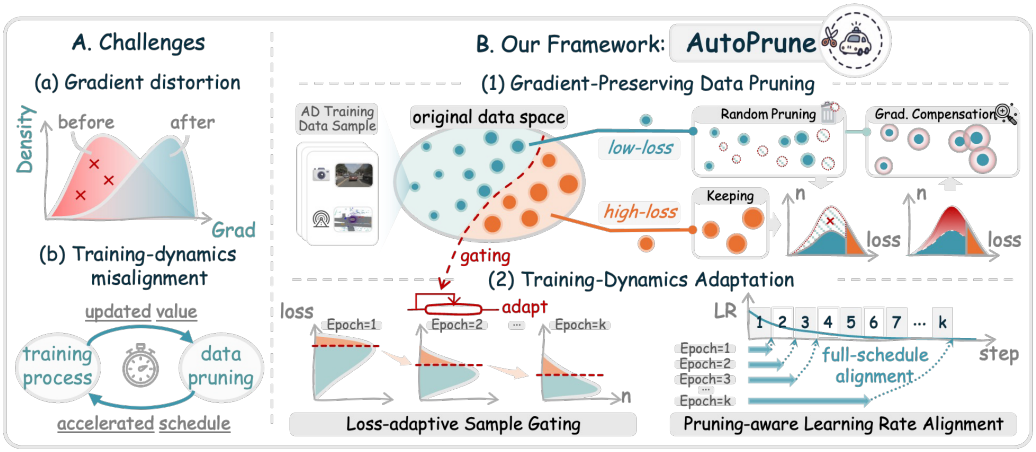

Figure 1: Overview of AutoPrune. The framework addresses gradient distortion and training-dynamics misalignment by (1) Gradient-Preserving Data Pruning, which randomly prunes low-loss samples and reweighting the retained samples to compensate for the omitted gradient contribution in expectation; and (2) Training-Dynamics Adaptation, which refreshes loss-based gates and aligning the learning-rate schedule with the reduced iteration budget.

signals, including model embeddings with density-aware clustering to preserve semantic diversity and trajectory entropy to maintain the behavioral distribution of planning data (Keser et al., 2025; Liu et al., 2025b). Although these methods improve data-value criteria, the subset remains fixed throughout training. As the model evolves, this static estimate may become increasingly inconsistent with its learning needs, leading to value staleness.

2.2 Dynamic Pruning Based on Model Feedback

Dynamic methods alleviate this staleness by periodically reassessing data value using feedback from the evolving model. Early approaches mainly used prediction uncertainty, task loss, and feature variation to identify informative frames or scenes for autonomous-driving perception (Das et al., 2020; 2021; Hekimoglu et al., 2022; Luo et al., 2023b). This model-feedback paradigm was later extended to prediction, planning, and end-to-end driving. For example, Lu et al. (2026) combine scene diversity with planning uncertainty and iteratively select data that are more valuable. Recent methods further broaden the selection signal: SEAD updates incremental training data using scene representations produced by the current model (Jiang et al., 2025a), while MOSAIC adjusts the composition of driving domains according to their estimated marginal contribution to planning performance (Dimlioglu et al., 2026). Although these methods allow data selection to evolve with the model, they primarily focus on determining which data should participate in training. The gradient distortion and training-dynamics misalignment introduced after data removal therefore remain insufficiently addressed.

3 Methodology

3.1 AutoPrune Overview

In this section, we introduce AutoPrune, a general training acceleration framework for autonomous driving. Figure 1 illustrates the overall workflow of AutoPrune. Gradient-Preserving Data Pruning removes part of the low-loss data and reweights the losses of the retained data to compensate for the gradient contributions omitted by pruning. To adapt this pruning process throughout training, loss-adaptive sample gating dynamically determines which data are protected or considered for pruning according to the evolving loss distribution, while pruning-aware learning-rate alignment synchronizes optimization progress with the corresponding stage of full-data training. The following subsections describe the two components in detail.

3.2 GRADIENT-PRESERVING DATA PRUNING

AutoPrune reduces the amount of data that actually participates in forward and backward propagation by pruning data that contribute less to the current model, thereby reducing the training burden. Implementing this process first requires answering a fundamental question: how should the contribution of training data to model learning be defined? Intuitively, the effect of training data is ultimately reflected in their influence on model parameter updates. Therefore, per-sample gradients provide the most direct signal for measuring training contributions. Let $\mathcal{S}_t$ denote the training samples used at step t. A general optimizer update is

$$\boldsymbol{\theta}_{t+1} = \mathcal{U}_t \left(\boldsymbol{\theta}_t, \sum_{i \in \mathcal{S}_t} \boldsymbol{g}_i^{(t)}, \eta_t \right), \tag{1}$$

where $\boldsymbol{\theta}_t$ and $\boldsymbol{\theta}_{t+1}$ are the model parameters before and after the update, $\boldsymbol{g}_i^{(t)}$ is the raw gradient induced by training sample i, η_t is the learning rate, and $\mathcal{U}_t$ is the optimizer-specific update operator.

However, individually computing, storing, and comparing the high-dimensional gradients associated with all training samples requires additional backward propagation, while the required storage grows jointly with the number of model parameters and the amount of training data. For massive autonomous driving training data, this introduces substantial computational, GPU memory, and storage overhead, contradicting the objective of accelerating training. Under a fixed model state $\boldsymbol{\theta}_t$, the per-sample gradient and loss satisfy

$$\boldsymbol{g}_i^{(t)} = \nabla_{\boldsymbol{\theta}} \ell_i(\boldsymbol{\theta}_t), \tag{2}$$

where $\ell_i(\boldsymbol{\theta}_t)$ denotes the training loss produced by the i-th sample under the model state $\boldsymbol{\theta}_t$, and $\nabla_{\boldsymbol{\theta}}$ denotes the gradient operator with respect to the model parameters. In contrast, per-sample losses can be obtained directly during normal forward propagation and training without introducing additional gradient computation. At a given model state, the per-sample loss quantifies each sample's scalar contribution to the empirical training objective. Samples with comparable objective-level contributions therefore exhibit comparable loss values and similar degrees of fit under the current model. Based on this relationship, we use per-sample loss as a low-cost signal to characterize their relative value under the current training state.

Let $\mathcal{D}$ denote the full training dataset, e denote the training round, and $\mathcal{L}_e$ denote the low-value data subset in the e-th round. The remaining data form the set $\mathcal{H}_e = \mathcal{D} \setminus \mathcal{L}_e$. Given a pruning ratio $\rho \in [0, 1)$ for low-value data, AutoPrune uniformly selects m_e samples without replacement from $\mathcal{L}_e$, forms the pruned subset $\mathcal{M}_e$ from the selected samples, and forms the retained subset $\mathcal{R}_e$ from the remaining samples:

$$m_e = \rho |\mathcal{L}_e|, \quad \mathcal{M}_e \sim \mathrm{Unif}\left\{ \mathcal{S} \subseteq \mathcal{L}_e \,|\, |\mathcal{S}| = m_e \right\}, \quad \mathcal{R}_e = \mathcal{L}_e \setminus \mathcal{M}_e. \tag{3}$$

Here, m_e denotes the number of samples pruned from the low-value set in the e-th round, $\mathcal{S}$ denotes any candidate subset of $\mathcal{L}_e$ containing m_e samples, and Unif denotes uniform random sampling over all such candidate subsets. Therefore, any sample $i \in \mathcal{L}_e$ is pruned with probability ρ and retained with probability $1 - \rho$. Let $\mathcal{A}_e$ denote the set of data that actually participates in training during the e-th round. Then,

$$\mathcal{A}_e = \mathcal{H}_e \cup \mathcal{R}_e. \tag{4}$$

In implementation, the first training round uses all data, i.e., $\mathcal{A}_1 = \mathcal{D}$, to obtain the losses of all training samples during the initial learning stage. $\ell_i^{(e)} = \ell_i(\boldsymbol{\theta}^{(e)})$ is used to denote the original loss actually computed for the i-th sample in the e-th round, where $\boldsymbol{\theta}^{(e)}$ denotes the model state corresponding to the e-th round; $\widetilde{\ell}_i^{(e)}$ denotes the latest available loss maintained for this sample after the e-th round. In subsequent rounds, unpruned samples update their loss records using their latest observed losses, whereas samples not participating in the current training round continue to use their most recently observed losses:

$$\widetilde{\ell}_i^{(e)} = \begin{cases} \ell_i^{(e)}, & i \in \mathcal{A}_e, \\ \widetilde{\ell}_i^{(e-1)}, & i \notin \mathcal{A}_e. \end{cases} \tag{5}$$

In this manner, without performing additional forward propagation for pruned data, AutoPrune can continuously obtain value signals for subsequent data selection by combining the latest losses of the data currently participating in training with the previously stored losses of the pruned data.

Although the data in $\mathcal{L}_e$ have relatively low losses, they still have nonzero gradients. Directly reducing these data inevitably weakens their contributions to cumulative parameter updates, thereby altering the original gradient composition of full-data training. Under the fixed model snapshot $\boldsymbol{\theta}^{(e)}$, let $\boldsymbol{g}_i^{(e)} = \nabla_{\boldsymbol{\theta}} \ell_i(\boldsymbol{\theta}^{(e)})$ denote the gradient produced by the i-th sample, and use $\boldsymbol{G}_e$ to denote the aggregate gradient formed by the full data:

$$\boldsymbol{G}_e = \sum_{i \in \mathcal{H}_e} \boldsymbol{g}_i^{(e)} + \sum_{i \in \mathcal{L}_e} \boldsymbol{g}_i^{(e)}. \tag{6}$$

If only the random subset $\mathcal{R}_e$ is retained without compensation, $\boldsymbol{G}'_e$ is used to denote the aggregate gradient formed after pruning, where the prime superscript distinguishes it from the full-data gradient $\boldsymbol{G}_e$:

$$\begin{aligned} \boldsymbol{G}'_e &= \sum_{i \in \mathcal{H}_e} \boldsymbol{g}_i^{(e)} + \sum_{i \in \mathcal{R}_e} \boldsymbol{g}_i^{(e)}, \\ \mathbb{E}_{\mathcal{M}_e}\left[\boldsymbol{G}'_e \mid \boldsymbol{\theta}^{(e)}, \mathcal{L}_e\right] &= \sum_{i \in \mathcal{H}_e} \boldsymbol{g}_i^{(e)} + (1-\rho) \sum_{i \in \mathcal{L}_e} \boldsymbol{g}_i^{(e)}. \end{aligned} \tag{7}$$

Here, $\mathbb{E}[\cdot]$ denotes the expectation over the sampling process of the random subset $\mathcal{R}_e$. Conditioning on $\boldsymbol{\theta}^{(e)}$ and $\mathcal{L}_e$ indicates that the expectation is taken only over the random pruning process while holding the model state and low-value data set fixed. Without compensation, the expected gradient contribution of the low-value data is reduced to $(1-\rho)$ times its original value, thereby disrupting the relative composition of different data groups within the original optimization signal.

To compensate for the missing training contributions of the pruned data, AutoPrune applies inverse-probability compensation to the losses of the randomly retained data according to the retention probability of low-value data. $\widehat{\ell}_i^{(e)}$ is used to denote the compensated per-sample loss, where the hat indicates that the loss has been compensated:

$$\widehat{\ell}_i^{(e)} = \begin{cases} \ell_i(\boldsymbol{\theta}^{(e)}), & i \in \mathcal{H}_e, \\ \dfrac{1}{1-\rho} \ell_i(\boldsymbol{\theta}^{(e)}), & i \in \mathcal{R}_e. \end{cases} \tag{8}$$

According to the differential relationship between the loss and gradient in Eq. (2), applying a constant compensation factor to the loss correspondingly adjusts the gradient produced through backward propagation:

$$\nabla_{\boldsymbol{\theta}} \widehat{\ell}_i^{(e)} = \begin{cases} \boldsymbol{g}_i^{(e)}, & i \in \mathcal{H}_e, \\ \dfrac{1}{1-\rho} \boldsymbol{g}_i^{(e)}, & i \in \mathcal{R}_e. \end{cases} \tag{9}$$

The aggregate gradient estimate formed by the compensated losses is further denoted as $\widehat{\boldsymbol{G}}_e$, where the hat indicates that the gradient includes compensation for the missing training contributions. Then,

$$\widehat{\boldsymbol{G}}_e = \sum_{i \in \mathcal{H}_e} \boldsymbol{g}_i^{(e)} + \frac{1}{1-\rho} \sum_{i \in \mathcal{R}_e} \boldsymbol{g}_i^{(e)}. \tag{10}$$

Given that each sample in the low-value set is retained with probability $1-\rho$, it follows that

$$\mathbb{E}_{\mathcal{M}_e}\left[\widehat{\boldsymbol{G}}_e \mid \boldsymbol{\theta}^{(e)}, \mathcal{L}_e\right] = \sum_{i \in \mathcal{H}_e} \boldsymbol{g}_i^{(e)} + \frac{1}{1-\rho} \sum_{i \in \mathcal{L}_e} (1-\rho) \boldsymbol{g}_i^{(e)} = \boldsymbol{G}_e. \tag{11}$$

The above analysis shows that inverse-probability reweighting mitigates the pruning-induced shift in the gradient distribution by making the reweighted aggregate gradient match its corresponding full-data counterpart in expectation over the pruning operation.

3.3 Training-Dynamics Adaptation

Building on the fixed-state gradient compensation above, AutoPrune further adapts data selection and optimization scheduling throughout training. As model parameters and per-sample losses evolve, data selection is updated to reflect the current learning state, while the learning-rate schedule is adjusted to account for the reduced iteration budget after pruning.

3.3.1 LOSS-ADAPTIVE SAMPLE GATING

Per-sample losses depend on the current model state. As training progresses, the model's degree of fit to different samples continuously changes, and the numerical scale and distribution of the overall losses change accordingly. Therefore, using a predefined and fixed loss value as the pruning threshold cannot maintain a consistent selection criterion across different training stages, resulting in erroneous and missed pruning.

Therefore, according to the current training state, AutoPrune deterministically retains a certain proportion of high-loss data and applies random pruning and gradient compensation only to the remaining data with relatively low losses.

Specifically, let $\gamma \in (0, 1)$ denote the proportion of high-loss data directly retained in each round. Before the e-th training round begins, AutoPrune computes a dynamic gating threshold according to the latest available losses maintained after the previous round, $\{\widetilde{\ell}_i^{(e-1)}\}_{i\in\mathcal{D}}$:

$$\tau_e = Q_{1-\gamma}\left(\left\{\widetilde{\ell}_i^{(e-1)}\right\}_{i\in\mathcal{D}}\right), \tag{12}$$

where τ_e denotes the loss-gating threshold for the e-th round, and $Q_{1-\gamma}(\cdot)$ denotes the $(1-\gamma)$-quantile of the input loss distribution. According to this threshold, the low-value set $\mathcal{L}_e$ and directly retained set $\mathcal{H}_e$ used in the previous section are specifically defined as

$$\mathcal{L}_e = \left\{i \in \mathcal{D} \mid \widetilde{\ell}_i^{(e-1)} < \tau_e\right\}, \quad \mathcal{H}_e = \left\{i \in \mathcal{D} \mid \widetilde{\ell}_i^{(e-1)} \geq \tau_e\right\}. \tag{13}$$

All high-loss data in $\mathcal{H}_e$ participate in the e-th training round. For the relatively low-loss data in $\mathcal{L}_e$, AutoPrune samples the pruned subset $\mathcal{M}_e$ according to the pruning ratio ρ defined in the previous section and retains $\mathcal{R}_e = \mathcal{L}_e \setminus \mathcal{M}_e$. The actual training set is therefore $\mathcal{A}_e = \mathcal{H}_e \cup \mathcal{R}_e$.

At the end of each training round, AutoPrune updates the latest available losses according to Eq. (5) and recomputes τ_{e+1} and the corresponding data sets in the next round. Therefore, the proportion of retained high-loss data controlled by γ remains fixed, whereas the specific gating threshold and the set to which each sample belongs are continuously updated with the model's learning state.

3.3.2 PRUNING-AWARE LEARNING RATE ALIGNMENT

Loss-gated pruning dynamically changes the amount of data participating in each training round, resulting in a smaller effective training set than full-data training. Since end-to-end autonomous driving models are commonly trained with iteration-based schedules, their learning-rate schedules are typically configured according to the full-data iteration budget. If the learning rate is still updated directly according to the actual iteration index after pruning, the reduced number of iterations causes a mismatch between the current training round and the optimization stage represented by the original learning-rate schedule.

To align these two progress measures, let B denote the global batch size. Completing one training round requires $T = \lceil |\mathcal{D}|/B \rceil$ iterations with the full data but only $T_e = \lceil |\mathcal{A}_e|/B \rceil$ iterations with the effective data set in the e-th round. AutoPrune organizes these T_e actual iterations into a dynamic training round, allowing the current effective data to be used approximately once. The resulting progress is then mapped back to the progress defined by full-data training. Let $u \in \{0, \ldots, T_e\}$ denote the number of actual iterations completed within the e-th round. The corresponding aligned iteration position is defined as

$$\bar{t}_{e,u} = (e-1)T + u\frac{T}{T_e}, \tag{14}$$

where $\bar{t}_{e,u}$ denotes the equivalent iteration position mapped to the full-data training progress. At the beginning of the e-th round, $\bar{t}_{e,0} = (e-1)T$; at the end of the round, $\bar{t}_{e,T_e} = eT$. Therefore, regardless of the actual number of iterations in the e-th round, its learning-rate progress corresponds to one complete round of full-data training.

Let $\Lambda(t)$ denote the learning-rate scheduling function configured for the original model under full-data training. The learning rate used at the u-th actual iteration of the e-th round is set to

$$\eta_{e,u} = \Lambda\left(\bar{t}_{e,u}\right), \tag{15}$$

Table 1: Main results under different pruning ratios. Percentages in parentheses denote changes relative to full-data training.

Overall Pruning Ratio	Method	L2 (m) ↓				Collision Rate (%) ↓				Speedup
		1s	2s	3s	Avg.	1s	2s	3s	Avg.	
0%	Full Data	0.29	0.56	0.93	0.59	0.00	0.05	0.21	0.09	1.00×
20%	Random	0.28	0.55	0.91	0.58 (-2.7%)	0.01	0.08	0.23	0.11 (+23.5%)	1.22×
	AutoPrune	0.27	0.53	0.90	0.57 **(-4.4%)**	0.00	0.03	0.19	0.07 **(-12.9%)**	
40%	Random	0.30	0.59	0.99	0.63 (+5.6%)	0.00	0.07	0.27	0.11 (+31.8%)	1.56×
	AutoPrune	0.29	0.56	0.94	0.59 **(+0.0%)**	0.00	0.03	0.22	0.08 **(-1.2%)**	
60%	Random	0.34	0.66	1.08	0.70 (+17.0%)	0.00	0.09	0.36	0.15 (+76.5%)	2.17×
	AutoPrune	0.30	0.59	1.00	0.63 **(+6.1%)**	0.00	0.08	0.30	0.13 **(+47.1%)**	

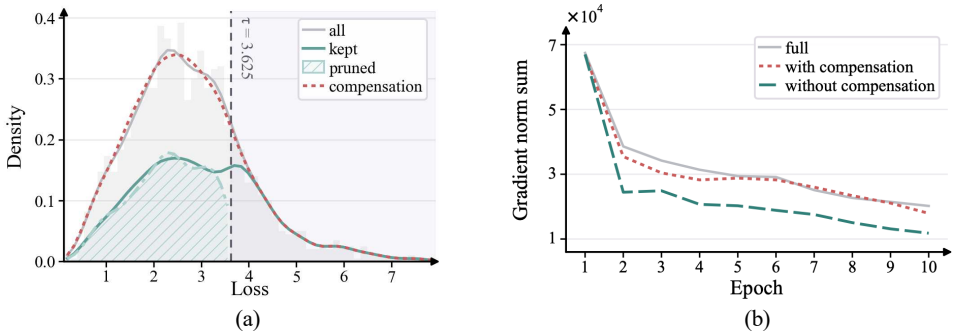

Figure 2: Illustration and empirical evaluation of the gradient-compensation process. (a) Loss-based pruning and inverse-probability weighting in the fourth epoch, where the weighted loss profile approximates the full-data reference. (b) Per-epoch sums of gradient norms, where compensation produces results closer to full-data training.

where $\eta_{e,u}$ denotes the aligned actual learning rate. This method reuses the original learning-rate scheduling function and queries it only at the equivalent iteration position $\bar{t}_{e,u}$.

Through the above mapping, each pruned training round traverses the same interval of the original optimization schedule as the corresponding full-data round, avoiding delayed schedule progression caused by the reduced number of iterations. Combined with gradient compensation, pruning-aware learning-rate alignment ensures that the reweighted gradients are used with learning rates indexed by the equivalent full-data progress, thereby coordinating gradient correction with schedule progression.

4 Experiments

4.1 Experimental Setup

Experiments are conducted on the standard nuScenes (Caesar et al., 2019) split, with the training set for model training and data pruning and the validation set for evaluation. AutoPrune is evaluated on representative end-to-end driving models, including SparseDrive (Sun et al., 2025), VAD (Jiang et al., 2023), DiffusionDrive (Liao et al., 2025), and DistillDrive (Yu et al., 2025). Pruning-ratio experiments, baseline comparisons, and ablations use the second training stage of SparseDrive-Small on four NVIDIA A800 GPUs; the other models assess applicability across architectures on four NVIDIA L40 GPUs. Established data-selection methods are adapted to the same end-to-end driving setting, including four static methods (CCS (Zheng et al., 2023), k-Center (Sener & Savarese, 2018), MaxHerding (Bae et al., 2025), and ProbCover (Yehuda et al., 2022)) and three dynamic methods (Dynamic Uncertainty (He et al., 2024), RHO-Loss (Mindermann et al., 2022), and UCB (Raju

Table 2: Results across different end-to-end driving models under an overall pruning ratio of 40%. Parenthetical percentages denote changes from the corresponding full-data reference.

Model	Method	L2 (m) ↓				Collision Rate (%) ↓			
		1s	2s	3s	Avg.	1s	2s	3s	Avg.
VAD	Full Data	0.56	0.88	1.25	0.89	0.22	0.45	0.85	0.51
	Random	0.51	0.84	1.23	0.86 (-3.8%)	0.62	0.79	1.10	0.84 (+64.2%)
	AutoPrune	0.46	0.77	1.14	0.79 **(-11.6%)**	0.26	0.41	0.70	0.46 **(-10.0%)**
DiffusionDrive	Full Data	0.31	0.61	1.02	0.65	0.00	0.05	0.19	0.08
	Random	0.38	0.71	1.13	0.74 (+14.2%)	0.00	0.04	0.20	0.08 (+1.4%)
	AutoPrune	0.32	0.63	1.01	0.65 **(+1.1%)**	0.02	0.05	0.18	0.08 **(+4.8%)**
DistillDrive	Full Data	0.27	0.53	0.89	0.56	0.03	0.08	0.19	0.10
	Random	0.28	0.55	0.92	0.58 (+3.5%)	0.02	0.06	0.26	0.11 (+14.1%)
	AutoPrune	0.27	0.54	0.90	0.57 **(+1.2%)**	0.00	0.05	0.23	0.10 **(-3.0%)**

et al., 2021)). By default, $\gamma = 20\%$, directly retaining the top 20% highest-loss samples; further implementation details are provided in Appendix A. Planning performance is measured using L2 displacement error and collision rate under the nuScenes open-loop evaluation.

4.2 Main Results

First, AutoPrune is compared with uniform random pruning in Table 1. Pruning ratios of $\rho = 25\%, 50\%, 75\%$ are applied to the low-loss candidates, which correspond to overall pruning ratios of 20%, 40%, and 60%, respectively, under $\gamma = 20\%$; the same setting is used in subsequent experiments unless otherwise specified. At a pruning ratio of 20%, random pruning substantially increases the collision rate by 23.5%, whereas AutoPrune maintains performance comparable to full-data training and even slightly reduces both the L2 error and collision rate. As the pruning ratio further increases to 40%, uniform random pruning suffers a 31.8% performance degradation, whereas AutoPrune remains on par with full-data training. Under the more aggressive pruning ratio of 60%, the substantial reduction in retained data limits the effectiveness of gradient compensation. Nevertheless, compared with uniform random pruning, AutoPrune narrows the gaps to full-data training by 11 and 29 percentage points in L2 error and collision rate, respectively. The SparseDrive results show that AutoPrune retains performance close to full-data training at pruning ratios of up to 40% and outperforms uniform random pruning at 60%. In particular, pruning 40% of the training data while maintaining comparable model performance enables a $1.56\times$ training speedup.

An intuitive illustration of the loss-based pruning and gradient-compensation process is provided in Figure 2. In Figure 2(a), the pruning of low-loss samples in the fourth epoch is visualized, and it is shown that inverse-probability weighting of the retained samples produces a weighted loss profile close to the full-data reference, providing intuition for the reweighting design. In Figure 2(b), the per-epoch sum of raw gradient norms is reported. With compensation, this magnitude-level statistic remains closer to the full-data reference than without compensation, providing empirical support for the intended gradient-compensation effect.

Furthermore, the cross-architecture applicability of AutoPrune is evaluated on VAD, DiffusionDrive, and DistillDrive in Table 2. For efficiency, the cross-architecture experiments use a fixed random 50% subset of the original training set for 10 epochs; within this protocol, full data denotes the complete selected subset, and both pruning methods use a 40% overall pruning ratio. On VAD, AutoPrune even surpasses full-data training, reducing the average L2 error and collision rate by 11.6% and 10.0%, respectively, whereas random pruning increases the collision rate by 64.2%. On DiffusionDrive, the two average metrics of AutoPrune change by only 1.1% and 4.8%, respectively, relative to full-data training and are identical when rounded to two decimal places, substantially mitigating the 14.2% degradation in L2 error caused by random pruning. On DistillDrive, AutoPrune limits the change in average L2 error to 1.2% while reducing the average collision rate by 3.0%; random pruning, in contrast, degrades the two metrics by 3.5% and 14.1%, respectively. These results support the applicability of AutoPrune across the evaluated architectures.

4.3 Comparison with Baselines

In Table 3, AutoPrune is compared with different data selection methods under the same overall pruning ratio of 40%. All static methods degrade the average L2 error by 2.5% to 6.1%, and their collision rates also increase except for CCS, suggesting that fixed coverage or representativeness criteria may not fully adapt to the model's evolving learning state. The dynamic methods are generally more competitive. Among them, UCB achieves an average L2 error of 0.61 m and a collision rate of 0.08%, showing competitive performance. While maintaining the same collision rate, AutoPrune further reduces the average L2 error to 0.59 m, matching full-data training. Although RHO-Loss dynamically exploits training signals, its collision rate increases by 47.1%, suggesting that dynamically adapting data selection alone may be insufficient to maintain model performance. Detailed results at each prediction horizon are provided in Appendix D.

Table 3: Data-selection baseline comparison at an overall 40% pruning ratio. Parenthetical percentages denote changes from full-data training.

Method	L2 Avg. (m) ↓	Col. Avg. (%) ↓
Baseline		
Full Data	0.59	0.09
Static		
CCS	0.63 (+6.1%)	0.09 (+0.0%)
k-Center	0.61 (+2.5%)	0.09 (+7.1%)
MaxHerding	0.63 (+5.2%)	0.11 (+24.7%)
ProbCover	0.62 (+4.4%)	0.11 (+25.9%)
Dynamic		
Dyn-Unc	0.61 (+3.0%)	0.09 (+10.6%)
RHO-Loss	0.62 (+4.4%)	0.13 (+47.1%)
UCB	0.61 (+1.9%)	0.08 (-1.2%)
Ours		
AutoPrune	**0.59 (+0.0%)**	**0.08 (-1.2%)**

4.4 Ablation Study

Using the complete AutoPrune as the reference, the contribution of each functional module under an overall pruning ratio of 40% is analyzed in Table 4. After removing loss-adaptive sample gating, the average L2 error and collision rate increase to 0.60 m (+0.5%) and 0.11% (+28.6%), respectively, relative to the complete method, supporting the benefit of updating the protected high-loss set using current loss records, particularly for collision-related performance.

Table 4: Ablation study at an overall pruning ratio of 40%. LG, LR, and GC denote loss-adaptive gating, learning-rate alignment, and gradient compensation, respectively. Parenthetical percentages denote changes from complete AutoPrune.

LG	LR	GC	L2 Avg. (m) ↓	Col. Avg. (%) ↓
-	-	-	0.63 (+5.6%)	0.11 (+33.3%)
-	✓	✓	0.60 (+0.5%)	0.11 (+28.6%)
✓	✓	-	0.61 (+3.0%)	0.09 (+10.7%)
✓	-	✓	0.63 (+5.9%)	0.10 (+21.4%)
✓	✓	✓	**0.59**	**0.08**

After removing gradient compensation, the average L2 error and collision rate increase by 3.0% and 10.7%, respectively, supporting that compensating for the missing training contributions of pruned data can mitigate performance degradation. After removing learning-rate alignment, the two metrics degrade by 5.9% and 21.4%, respectively, with the degradation in L2 error being the most pronounced, supporting the benefit of aligning the optimization schedule under pruning with the corresponding full-data training progress. When all three modules are removed, performing only random data pruning degrades the average L2 error and collision rate by 5.6% and 33.3%, respectively. Also, the corresponding horizon-wise results are reported in Appendix D.

5 Conclusion

This paper presents AutoPrune, an adaptive data-pruning framework for accelerating end-to-end driving training. Rather than treating data pruning solely as sample selection, AutoPrune combines random pruning of low-loss candidates with inverse-probability gradient correction and training-dynamics adaptation. The reweighted aggregate gradient matches its full-data counterpart in expectation over random pruning, while loss-adaptive gating and pruning-aware learning-rate alignment adapt data selection and schedule progression throughout training. Experiments on SparseDrive show that AutoPrune achieves performance comparable to that of full-data training with an overall pruning ratio of 40%. Comparisons with existing selection methods and evaluations across additional architectures further support its empirical effectiveness and cross-architecture applicability.

AI USE STATEMENT

Generative AI tools were used to assist with manuscript editing, including language polishing, formatting, and consistency checks. They were not used to generate experimental data or results. All AI-assisted content was reviewed and verified by the authors, who take responsibility for the final manuscript.

ETHICS STATEMENT

This work studies training-data pruning for autonomous-driving models using a public benchmark. It involves no human-subject experiments or deployment on public roads; any real-world use would require independent closed-loop testing and safety validation before deployment.

REPRODUCIBILITY STATEMENT

The experimental setup, principal hyperparameters, and evaluation protocol are provided in the main paper and Appendix A. The evaluated architectures, baseline implementation details, and additional results are provided in Appendices B–D. Our anonymous repository at `https://anonymous.4open.science/r/AutoPrune-18AF/` provides the model-independent reference implementation of AutoPrune, hyperparameter configurations, training and evaluation tools, the adapter interface, and method-level tests.

REFERENCES

Wonho Bae, Junhyug Noh, and Danica J Sutherland. Generalized coverage for more robust low-budget active learning. In *18th European Conference on Computer Vision, ECCV 2024*, pp. 318–334. Springer Science and Business Media Deutschland GmbH, 2025.

Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. *arXiv preprint arXiv:1903.11027*, 2019.

Soumi Das, Sayan Mandal, Ashwin Bhoyar, Madhumita Bharde, Niloy Ganguly, Suparna Bhattacharya, and Sourangshu Bhattacharya. Multi-criteria online frame-subset selection for autonomous vehicle videos. *Pattern Recognition Letters*, 133:349–355, 2020.

Soumi Das, Harikrishna Patibandla, Suparna Bhattacharya, Kshounis Bera, Niloy Ganguly, and Sourangshu Bhattacharya. Tmcoss: Thresholded multi-criteria online subset selection for data-efficient autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6341–6350, 2021.

Tolga Dimlioglu, Nadine Chang, Maying Shen, Rafid Mahmood, and Jose M Alvarez. Scaling-aware data selection for end-to-end autonomous driving systems. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17798–17808, 2026.

Ross Greer and Mohan M Trivedi. Perception without vision for trajectory prediction: Ego vehicle dynamics as scene representation for efficient active learning in autonomous driving. In *2025 IEEE 28th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 3022–3029. IEEE, 2025.

Muyang He, Shuo Yang, Tiejun Huang, and Bo Zhao. Large-scale dataset pruning with dynamic uncertainty. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, CVPRW 2024*, pp. 7713–7722. IEEE Computer Society, 2024.

Aral Hekimoglu, Michael Schmidt, Alvaro Marcos-Ramiro, and Gerhard Rigoll. Efficient active learning strategies for monocular 3d object detection. In *2022 IEEE Intelligent Vehicles Symposium (IV)*, pp. 295–302. IEEE, 2022.

Bo Jiang, Shaoyu Chen, Qing Xu, Bencheng Liao, Jiajie Chen, Helong Zhou, Qian Zhang, Wenyu Liu, Chang Huang, and Xinggang Wang. Vad: Vectorized scene representation for efficient autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8340–8350, 2023.

Wenhao Jiang, Duo Li, Menghan Hu, Chao Ma, Ke Wang, and Zhipeng Zhang. Active learning from scene embeddings for end-to-end autonomous driving. *arXiv preprint arXiv:2503.11062*, 2025a.

Xiyan Jiang, Xiaocong Zhao, Yiru Liu, Zirui Li, Peng Hang, Lu Xiong, and Jian Sun. A naturalistic trajectory dataset with dense interaction for autonomous driving. *Scientific Data*, 12(1):1084, 2025b.

Mert Keser, Sinan Simsek, Deniz Karacor, and Alois Knoll. Subset selection for autonomous driving datasets via fine-tuning vision foundation models. In *2025 IEEE Intelligent Vehicles Symposium (IV)*, pp. 2251–2258. IEEE, 2025.

Chenyang Lei, Weiyuan Peng, Guang Zhou, Meiying Zhang, Qi Hao, Chunlin Ji, and Chengzhong Xu. Tscenejal: Joint active learning of traffic scenes for 3d object detection. *IEEE Transactions on Intelligent Transportation Systems*, 2025.

Bencheng Liao, Shaoyu Chen, Haoran Yin, Bo Jiang, Cheng Wang, Sixu Yan, Xinbang Zhang, Xiangyu Li, Ying Zhang, Qian Zhang, et al. Diffusiondrive: Truncated diffusion model for end-to-end autonomous driving. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 12037–12047, 2025.

Chang Liu, Mingxu Zhu, Zheyuan Zhang, Linna Song, Xiao Zhao, Qingliang Luo, Qi Wang, Chufan Guo, and Kuifeng Su. Tad-e2e: A large-scale end-to-end autonomous driving dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 26600–26609, 2025a.

Zhaoyang Liu, Weitao Zhou, Junze Wen, Cheng Jing, Qian Cheng, Kun Jiang, and Diange Yang. Are all data necessary? efficient data pruning for large-scale autonomous driving dataset via trajectory entropy maximization. *arXiv preprint arXiv:2512.19270*, 2025b.

Han Lu, Xiaosong Jia, Yichen Xie, Siyu Sun, Wenlong Liao, Xiaokang Yang, and Junchi Yan. Activead: Planning-oriented active learning for end-to-end autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3656–3666, 2026.

Yadan Luo, Zhuoxiao Chen, Zhen Fang, Zheng Zhang, Mahsa Baktashmotlagh, and Zi Huang. Kecor: Kernel coding rate maximization for active 3d object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 18279–18290, 2023a.

Yadan Luo, Zhuoxiao Chen, Zijian Wang, Xin Yu, Zi Huang, and Mahsa Baktashmotlagh. Exploring active 3d object detection from a generalization perspective. *arXiv preprint arXiv:2301.09249*, 2023b.

Sören Mindermann, Jan M Brauner, Muhammed T Razzak, Mrinank Sharma, Andreas Kirsch, Winnie Xu, Benedikt Höltgen, Aidan N Gomez, Adrien Morisot, Sebastian Farquhar, et al. Prioritized training on points that are learnable, worth learning, and not yet learnt. In *International Conference on Machine Learning*, pp. 15630–15649. PMLR, 2022.

Alexander Naumann, Xunjiang Gu, Tolga Dimlioglu, Mariusz Bojarski, Alperen Degirmenci, Alexander Popov, Devansh Bisla, Marco Pavone, Urs Muller, and Boris Ivanovic. Data scaling laws for end-to-end autonomous driving. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 2571–2582, 2025.

Ravi S Raju, Kyle Daruwalla, and Mikko Lipasti. Accelerating deep learning with dynamic data pruning. *arXiv preprint arXiv:2111.12621*, 2021.

Abbas Sadat, Sean Segal, Sergio Casas, James Tu, Bin Yang, Raquel Urtasun, and Ersin Yumer. Diverse complexity measures for dataset curation in self-driving. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8609–8616. IEEE, 2021.

Sean Segal, Nishanth Kumar, Sergio Casas, Wenyuan Zeng, Mengye Ren, Jingkang Wang, and Raquel Urtasun. Just label what you need: Fine-grained active selection for p&p through partially labeled scenes. In *Conference on Robot Learning*, pp. 816–826. PMLR, 2022.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018.

Wenchao Sun, Xuewu Lin, Yining Shi, Chuang Zhang, Haoran Wu, and Sifa Zheng. Sparsedrive: End-to-end autonomous driving via sparse scene representation. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8795–8801. IEEE, 2025.

Ruining Yang and Lili Su. Data-efficient trajectory prediction via coreset selection. *arXiv preprint arXiv:2409.17385*, 2024.

Ofer Yehuda, Avihu Dekel, Guy Hacohen, and Daphna Weinshall. Active learning through a covering lens. *Advances in Neural Information Processing Systems*, 35:22354–22367, 2022.

Rui Yu, Xianghang Zhang, Runkai Zhao, Huaicheng Yan, and Meng Wang. Distilldrive: End-to-end multi-mode autonomous driving distillation by isomorphic hetero-source planning model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 26188–26197, 2025.

Haizhong Zheng, Rui Liu, Fan Lai, and Atul Prakash. Coverage-centric coreset selection for high pruning rates. In *11th International Conference on Learning Representations, ICLR 2023*, 2023.

A. Algorithm Procedure and Implementation Details

A.1 Algorithm Procedure

Although the individual components of AutoPrune have been detailed in the methodology, their execution order and interaction throughout the complete training process are further clarified here. To this end, Algorithm A.1 summarizes the round-wise implementation of AutoPrune, including full-data initialization, loss-adaptive gating, random pruning of low-loss samples, gradient compensation, learning-rate alignment, and loss-record updating.

A.2 Implementation Details

AutoPrune introduces only two independent hyperparameters: the direct retention ratio γ for high-loss samples and the pruning ratio ρ within the low-loss candidate set. Specifically, γ determines the proportion of high-loss data that is directly retained in each training round, while ρ controls the proportion randomly removed from the remaining low-loss candidates. The gating threshold, gradient-compensation factor, and overall pruning ratio are all derived from these two parameters and require no additional tuning.

The high-loss retention ratio is set to $\gamma = 20\%$ by default, based jointly on the target pruning range and the feasibility of gradient compensation. Ignoring loss ties at the quantile boundary and integer rounding, a proportion γ of the data is assigned to the directly retained high-loss set, while the remaining proportion $1-\gamma$ forms the low-loss candidate set. Accordingly, the overall pruning ratio p is given by

$$p = (1-\gamma)\rho, \tag{16}$$

where ρ denotes the pruning ratio within the low-loss candidate set.

Our experiments evaluate overall pruning ratios of 20%, 40%, and 60% under a unified setting. Therefore, the low-loss candidate set must cover at least 60% of the full dataset. However, if low-loss candidates account for exactly 60% of the data, achieving an overall pruning ratio of 60% would require $\rho = 1$, meaning that all low-loss candidates would be removed. In this case, no low-loss samples would remain to compensate for the omitted gradient contributions, and the inverse-probability compensation factor $1/(1-\rho)$ would become undefined. The proportion of low-loss candidates must therefore be greater than the maximum target pruning ratio, ensuring that some samples with similar training behavior remain available for gradient compensation even under aggressive pruning.

Based on these considerations, γ is set to 20%, leaving the remaining 80% of the data as low-loss candidates. This setting directly protects high-loss samples while providing a sufficiently large candidate set for different pruning levels. Even at an overall pruning ratio of 60%, low-loss samples corresponding to 20% of the full dataset remain active for compensated training, ensuring that inverse-probability compensation remains feasible.

With $\gamma = 20\%$, the overall pruning ratio becomes

$$p = 0.8\rho. \tag{17}$$

Therefore, overall pruning ratios of 20%, 40%, and 60% are obtained by setting the pruning ratio ρ within the low-loss candidate set to 25%, 50%, and 75%, respectively:

$$\begin{aligned} \rho = 25\% &\Rightarrow p = 80\% \times 25\% = 20\%, \\ \rho = 50\% &\Rightarrow p = 80\% \times 50\% = 40\%, \\ \rho = 75\% &\Rightarrow p = 80\% \times 75\% = 60\%. \end{aligned} \tag{18}$$

The corresponding inverse-probability compensation factors $1/(1-\rho)$ for the retained low-loss samples are $4/3$, 2, and 4, respectively. Thus, different overall pruning ratios can be conveniently controlled by adjusting ρ, while a nonempty subset of low-loss data is retained under every setting to compensate for the gradient contributions omitted by pruning.

B. Evaluated End-to-End Driving Models

To evaluate the applicability of AutoPrune across different end-to-end autonomous driving architectures, SparseDrive, VAD, DiffusionDrive, and DistillDrive are selected as the evaluated models.

Algorithm A.1 Training Procedure of AutoPrune

Require: Full training dataset $\mathcal{D}$; total training rounds E; global batch size B; high-loss retention ratio γ; low-loss pruning ratio ρ; original learning-rate schedule $\Lambda(\cdot)$; initial model parameters $\boldsymbol{\theta}$

Ensure: Trained model parameters $\boldsymbol{\theta}$

$T \leftarrow \lceil |\mathcal{D}|/B \rceil$

The first round uses all samples to initialize loss records.

for $e = 1, \ldots, E$ **do**

  if $e = 1$ **then**

    $\mathcal{A}_1 \leftarrow \mathcal{D}$

  else

    Update the loss gate using the latest available loss records.

    $\tau_e \leftarrow Q_{1-\gamma}\left(\left\{\widetilde{\ell}_i^{(e-1)}\right\}_{i\in\mathcal{D}}\right)$

    $\mathcal{L}_e \leftarrow \left\{i \in \mathcal{D} \mid \widetilde{\ell}_i^{(e-1)} < \tau_e\right\}$

    $\mathcal{H}_e \leftarrow \left\{i \in \mathcal{D} \mid \widetilde{\ell}_i^{(e-1)} \geq \tau_e\right\}$

    Directly retain high-loss samples and randomly prune low-loss candidates.

    $m_e \leftarrow \rho|\mathcal{L}_e|$

    $\mathcal{M}_e \sim \mathrm{Unif}\{\mathcal{S} \subseteq \mathcal{L}_e \mid |\mathcal{S}| = m_e\}$

    $\mathcal{R}_e \leftarrow \mathcal{L}_e \setminus \mathcal{M}_e$

    $\mathcal{A}_e \leftarrow \mathcal{H}_e \cup \mathcal{R}_e$

  end if

  Determine the effective iteration budget after pruning.

  $T_e \leftarrow \lceil |\mathcal{A}_e|/B \rceil$

  Organize $\mathcal{A}_e$ into T_e mini-batches $\{\mathcal{S}_{e,u}\}_{u=0}^{T_e-1}$

  for $u = 0, \ldots, T_e - 1$ **do**

    Align the current iteration with full-data training progress.

    $\bar{t}_{e,u} \leftarrow (e-1)T + u\dfrac{T}{T_e}$

    $\eta_{e,u} \leftarrow \Lambda\left(\bar{t}_{e,u}\right)$

    Compensate the retained low-loss samples during training.

    for $i \in \mathcal{S}_{e,u}$ **do**

      Compute the original loss $\ell_i^{(e)} \leftarrow \ell_i(\boldsymbol{\theta})$

      if $e = 1$ or $i \in \mathcal{H}_e$ **then**

        $\widehat{\ell}_i^{(e)} \leftarrow \ell_i^{(e)}$

      else

        $\widehat{\ell}_i^{(e)} \leftarrow \dfrac{1}{1-\rho}\ell_i^{(e)}$

      end if

    end for

    $\boldsymbol{\theta} \leftarrow \mathcal{U}\left(\boldsymbol{\theta}, \sum_{i\in\mathcal{S}_{e,u}} \nabla_{\boldsymbol{\theta}}\widehat{\ell}_i^{(e)}, \eta_{e,u}\right)$

  end for

  Update observed losses while retaining the latest records of pruned samples.

  for $i \in \mathcal{D}$ **do**

    if $i \in \mathcal{A}_e$ **then**

      $\widetilde{\ell}_i^{(e)} \leftarrow \ell_i^{(e)}$

    else

      $\widetilde{\ell}_i^{(e)} \leftarrow \widetilde{\ell}_i^{(e-1)}$

    end if

  end for

end for

return $\boldsymbol{\theta}$

These models differ in their scene representations, planning generation mechanisms, and training objectives, allowing AutoPrune to be examined without relying on a specific model architecture.

SparseDrive. SparseDrive (Sun et al., 2025), presented at ICRA 2025, is an end-to-end autonomous driving framework based on sparse scene representations. Its symmetric sparse perception module represents object detection, tracking, and online mapping through sparse instances, while

Table A.1: Comparison with static and dynamic data selection baselines under an overall pruning ratio of 40%. Percentages denote changes relative to full-data training.

Type	Method	L2 (m) ↓				Collision Rate (%) ↓			
		1s	2s	3s	Avg.	1s	2s	3s	Avg.
Baseline	Full Data	0.29	0.56	0.93	0.59	0.00	0.05	0.21	0.09
Static	CCS	0.30	0.60	0.99	0.63 (+6.1%)	0.00	0.04	0.21	0.09 (+0.0%)
	k-Center	0.29	0.58	0.96	0.61 (+2.5%)	0.00	0.03	0.24	0.09 (+7.1%)
	MaxHerding	0.30	0.59	0.98	0.63 (+5.2%)	0.00	0.05	0.26	0.11 (+24.7%)
	ProbCover	0.30	0.59	0.98	0.62 (+4.4%)	0.01	0.06	0.25	0.11 (+25.9%)
Dynamic	Dyn-Unc	0.29	0.58	0.97	0.61 (+3.0%)	0.00	0.04	0.24	0.09 (+10.6%)
	RHO-Loss	0.30	0.59	0.98	0.62 (+4.4%)	0.01	0.07	0.30	0.13 (+47.1%)
	UCB	0.29	0.57	0.95	0.61 (+1.9%)	0.00	0.03	0.22	0.08 (-1.2%)
Ours	**AutoPrune**	0.29	0.56	0.94	**0.59 (+0.0%)**	0.00	0.03	0.22	**0.08 (-1.2%)**

its parallel motion planner simultaneously performs motion prediction for surrounding agents and trajectory planning for the ego vehicle. SparseDrive adopts a two-stage training procedure, and the second-stage training of SparseDrive-Small is used for the main pruning-ratio experiments, baseline comparisons, and ablation studies, making it the primary model for evaluating the individual components of AutoPrune.

VAD. VAD (Jiang et al., 2023), presented at ICCV 2023, is an end-to-end autonomous driving framework based on a fully vectorized scene representation. Agent motions and map elements are represented as structured vectors and used as instance-level planning constraints, avoiding computationally intensive rasterized scene representations and hand-crafted planning post-processing. Because its scene organization and planning constraints differ substantially from those of SparseDrive, VAD is used to examine the applicability of AutoPrune to vectorized end-to-end driving architectures.

DiffusionDrive. DiffusionDrive (Liao et al., 2025), presented at CVPR 2025, is an end-to-end autonomous driving framework built upon a truncated diffusion policy. Multi-modal trajectory anchors are used to construct an anchored Gaussian distribution, and a cascaded diffusion decoder learns the denoising process from this distribution to the multi-modal driving trajectory distribution, reducing the required diffusion steps while retaining diverse trajectory generation. As a representative generative planning paradigm, DiffusionDrive is used to evaluate whether AutoPrune can be applied to driving models with diffusion-based optimization objectives and multi-modal trajectory generation.

DistillDrive. DistillDrive (Yu et al., 2025), presented at ICCV 2025, is a knowledge-distillation-based framework for multi-modal end-to-end autonomous driving. A planning model built upon structured scene representations is employed as the teacher, and its diverse planning instances are used to supervise multi-modal motion feature learning in the end-to-end model, while reinforcement learning and generative modeling further enhance state-to-decision mappings and latent-space instance interactions. By introducing teacher-derived planning supervision beyond conventional training objectives, DistillDrive is used to further assess the applicability of AutoPrune to distillation-based end-to-end driving training.

C. Baseline Methods

To comprehensively evaluate AutoPrune, several existing data-selection methods are adapted to the same end-to-end autonomous driving training pipeline. These methods include four static approaches, namely CCS, k-Center, MaxHerding, and ProbCover, and three dynamic approaches, namely Dynamic Uncertainty, RHO-Loss, and UCB. All methods are evaluated using the same overall pruning ratio and base-model training configuration, enabling a controlled comparison of

Table A.2: Ablation study under an overall pruning ratio of 40%. LG, LR, and GC denote loss-adaptive gating, learning-rate alignment, and gradient compensation respectively. Percentages denote changes relative to complete AutoPrune.

LG	LR	GC	L2 (m) ↓				Collision Rate (%) ↓			
			1s	2s	3s	Avg.	1s	2s	3s	Avg.
-	-	-	0.30	0.59	0.99	0.63 (+5.6%)	0.00	0.07	0.27	0.11 (+33.3%)
-	✓	✓	0.29	0.56	0.94	0.60 (+0.5%)	0.00	0.05	0.27	0.11 (+28.6%)
✓	✓	-	0.29	0.58	0.96	0.61 (+3.0%)	0.00	0.04	0.23	0.09 (+10.7%)
✓	-	✓	0.31	0.60	0.98	0.63 (+5.9%)	0.00	0.05	0.26	0.10 (+21.4%)
✓	✓	✓	0.29	0.56	0.94	0.59	0.00	0.03	0.22	0.08

different data-selection mechanisms for end-to-end driving training. While preserving the central selection criterion of each method, we further adapt its input representation, scoring signal, and update procedure as consistently as possible to the multi-task training process of SparseDrive.

CCS. Coverage-centric Coreset Selection (CCS) (Zheng et al., 2023), published at ICLR 2023, was originally developed for one-shot coreset selection under high pruning ratios in general deep learning. Because conventional importance-based selection methods may suffer from insufficient data coverage and substantial performance degradation under aggressive pruning, CCS jointly considers sample importance and distributional coverage when constructing a fixed training subset. In our implementation, each training sample is represented by a nuScenes metadata feature vector containing temporal information, ego-vehicle state and pose, and the class counts and bounding-box statistics of annotated objects. Each feature dimension is standardized, and the resulting vector is ℓ_2-normalized before selection; the same representation is used throughout the static baselines below. Because these metadata do not provide a separate pretrained importance score, cosine similarity between each sample feature and the global feature mean is used as sample importance, and the greedy criterion assigns equal weight to normalized importance and the minimum cosine distance to the current selected set.

k-Center. k-Center (Sener & Savarese, 2018) originates from the core-set active learning method published at ICLR 2018, which was designed to reduce annotation requirements for image classification models. It formulates active learning as a core-set selection problem and greedily selects samples farthest from the existing centers in the feature space, allowing the selected subset to cover the full data distribution. In our implementation, it operates on the standardized and ℓ_2-normalized metadata feature vectors defined above and performs farthest-first selection using cosine distance. The selected set is initialized with the representative sample most similar to the global feature mean, after which the sample with the largest minimum distance to the existing centers is repeatedly added to form a fixed subset for stage-two training.

MaxHerding. MaxHerding (Bae et al., 2025), published at ECCV 2024, was originally proposed for image active learning under low annotation budgets. To address the sensitivity of ProbCover to its coverage-radius hyperparameter, MaxHerding introduces a smoother generalized coverage objective and employs a greedy selection strategy closely related to kernel herding. In our implementation, a kernel-herding-style criterion is applied to the normalized metadata feature vectors to approximate the mean representation of the complete dataset. A candidate is scored by its similarity to the global feature mean minus its average similarity to the selected samples, favoring driving data that represent the overall distribution without being overly redundant with the current subset.

ProbCover. ProbCover (Yehuda et al., 2022), published at NeurIPS 2022, was developed for deep active learning in the low-budget regime. It formulates data selection through probability coverage and prioritizes samples that cover a large number of previously uncovered points in the representation space, allowing a small labeled subset to preserve the representativeness of the original data distribution. In our implementation, local coverage relationships are constructed in the cosine-distance space of the normalized metadata feature vectors. The coverage radius is determined automati-

cally using the median neighborhood distance estimated from at most 1,024 uniformly sampled data points, after which candidates covering the largest number of uncovered samples are selected greedily. If all samples become covered early, the remaining slots are filled according to the k-Center ordering.

Dynamic Uncertainty. Dynamic Uncertainty (He et al., 2024), published at the CVPR Workshops 2024, was designed for training-data pruning on large-scale image-classification datasets. Instead of relying on uncertainty measured at a single training stage, it characterizes dynamic uncertainty through variations in model predictions throughout training, thereby distinguishing informative samples from easy-to-learn data and potential noise. To adapt it to SparseDrive, we use the per-sample loss defined in the main paper as the dynamic signal; this loss is computed from SparseDrive's original detection, mapping, motion-prediction, planning, and auxiliary-depth training objectives. We maintain each sample's latest loss, previous observed loss, and online variance. The score is the sum of the absolute loss change and running standard deviation; a cold-start sample observed only once uses its current loss, and the training subset is refreshed from the updated scores at each virtual-epoch boundary.

RHO-Loss. RHO-Loss (Mindermann et al., 2022), published at ICML 2022, was introduced to reduce the computation spent on redundant, already learned, or unlearnable data during large-scale model training. It uses reducible holdout loss to estimate the potential contribution of each sample to model generalization and prioritizes samples that are learnable, worth learning, and not yet sufficiently learned, rather than merely selecting difficult high-loss samples. Because SparseDrive combines structured objectives for detection, mapping, motion prediction, planning, and auxiliary depth supervision, training an additional independent holdout model would introduce a difficult-to-align task definition and substantial auxiliary cost. We therefore use an online adaptation in which the minimum per-sample loss observed so far for each sample serves as its reference, and the difference between the latest available per-sample loss and this historical minimum defines the reducible-loss score used to refresh the subsequent training subset.

UCB. The UCB baseline is derived from the 2021 preprint *Accelerating Deep Learning with Dynamic Data Pruning* (Raju et al., 2021), which formulates dynamic data pruning as an online decision-making problem coupled with the model training trajectory. Its UCB strategy combines the running mean of sample scores with an exploration bonus, exploiting currently high-value data while prioritizing samples whose values remain insufficiently estimated. To adapt it to SparseDrive, each training sample is treated as an arm, with its observation count and running mean maintained from its per-sample loss. The score is the running mean plus $c\sqrt{\log(N+1)/n_i}$, where n_i is the observation count of sample i, N is the total number of observations, and $c = 1$ in our experiments; the training subset is refreshed round by round according to the updated UCB scores.

These methods cover representative data-selection mechanisms based on importance and distributional coverage, feature-space coresets, probability coverage, training dynamics, uncertainty, and online decision making. They have been extensively validated in active learning, image classification, or general dataset-pruning tasks. Under a unified end-to-end driving training configuration and pruning budget, the favorable performance of AutoPrune against these competitive static and dynamic baselines provides further evidence for the effectiveness of combining gradient compensation with training-dynamics adaptation for driving-data pruning.

D. Additional Experimental Results

D.1 Detailed Quantitative Results

The main paper reports only the average planning metrics because of space constraints. Tables A.1 and A.2 provide the complete L2 displacement errors and collision rates at the 1 s, 2 s, and 3 s prediction horizons for the baseline comparison and ablation study, respectively.

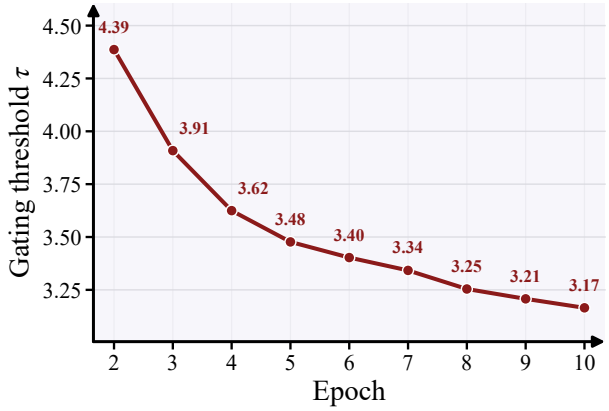

Figure A.1: Evolution of the loss-gating threshold throughout training. With $\gamma = 20\%$, τ_e is determined by the 80th percentile of the latest available sample-loss records at the beginning of epoch e.

D.2 Loss-Gating Dynamics

The loss-gating threshold exhibits clear stage-dependent evolution throughout training, as shown in Figure A.1. With $\gamma = 20\%$, τ_e is determined by the 80th percentile of the latest available sample-loss records, and its variation therefore reflects how the statistical scale of the overall loss distribution evolves with the model learning state. The threshold continuously decreases from 4.39 in the second epoch to 3.17 in the tenth epoch, with a sharper decline in the early stage and gradual stabilization thereafter. Consequently, the same absolute loss value does not retain a consistent gating meaning across training stages, making a fixed threshold unsuitable for identifying a consistent data range throughout training. By recomputing τ_e from the latest loss records at each epoch, loss-adaptive sample gating allows the directly retained high-loss range to follow the evolving training state while maintaining the predefined high-loss retention ratio.

E. Limitations and Future Directions

Although AutoPrune reduces training-data usage while maintaining performance close to full-data training, several limitations remain and motivate further investigation.

- **The gating range and pruning strength are not yet jointly adaptive.** While the gating threshold evolves with the loss distribution, the high-loss retention ratio and low-loss pruning ratio remain predefined. Future work may jointly adapt them to the training state for a better balance between training efficiency and model performance.
- **Gradient preservation remains limited to expectation-level aggregate consistency.** The current method primarily preserves consistency between the compensated and full-data gradients at the expectation level. However, gradient direction directly determines the actual direction of parameter updates, and expectation-level aggregate consistency alone cannot fully characterize the effect of pruning on optimization. Future work may establish finer-grained theoretical characterizations of gradient direction, gradient covariance, and optimization trajectories, and develop compensation mechanisms that jointly constrain gradient magnitude and directional deviation. This may mitigate gradient deviation and variance amplification under aggressive pruning, enabling more stable gradient recovery and performance compensation at higher pruning ratios.
- **Cross-dataset generalization has not been sufficiently validated.** The current experiments demonstrate the applicability of AutoPrune to different end-to-end driving models on nuScenes, but do not yet cover different datasets. Future work may further evaluate its generalization and training acceleration on additional large-scale autonomous-driving datasets.

Overall, these limitations are identified while recognizing the significance and demonstrated effectiveness of AutoPrune. Rather than diminishing its existing contributions, they delineate the current scope of the framework and indicate promising directions for further investigation.