

Article

PILOT: A Replay-Free Continual Learning Approach for Real-Time Semantic Segmentation via Boundary Guidance

Yujing Zhou ¹, Prashant Shekhar ^{1,*}, Thomas Yang ² and Yongxin Liu ^{1,*}¹ Department of Mathematics, College of Arts and Sciences, Embry-Riddle Aeronautical University, Daytona Beach, FL 32114, USA; zhouy9@my.erau.edu² Department of Electrical Engineering and Computer Science, College of Engineering, Embry-Riddle Aeronautical University, Daytona Beach, FL 32114, USA; yang482@erau.edu

* Correspondence: shekharp@erau.edu (P.S.); liuy11@erau.edu (Y.L.)

Abstract

Real-time semantic segmentation models offer an excellent balance between accuracy and inference speed. However, deploying these models in dynamic real-world environments often requires the ability to learn novel classes incrementally without retraining on the entire dataset. This capability is known as continual learning. In this regard, standard fine-tuning methods often suffer from catastrophic forgetting, where the model learns new information but loses accuracy on previously learned classes. The severity of this effect depends on the incremental setup, the available data, and the fine-tuning strategy. Contributing to this crucial domain, this paper proposes a novel continual learning framework tailored for PIDNet, which is a widely cited state-of-the-art real-time semantic segmentation model. Our method, PILOT (Parallel Incremental Learning Over Time), introduces a real-time and lightweight strategy by implementing a parallel Derivative branch (D-branch) designed to capture the high-frequency boundary information of novel classes while freezing the trained parameters of the original segmentation network. This novel setup allows the model to adapt to new semantic categories while preserving the knowledge of previously learned classes. By using only data associated with the new class, our model significantly reduces training overhead. Experimental results demonstrate that our approach successfully segments new classes while maintaining a high mean Intersection over Union (mIoU) on the original base classes, thereby outperforming prior continual learning approaches in this real-time segmentation setting. Overall, PILOT is shown to effectively mitigate catastrophic forgetting with minimal impact on inference latency, adding fewer than 5% additional parameters and reducing the frame rate by only about 9%, thus maintaining real-time performance.

Keywords: continual learning; real-time semantic segmentation; class-incremental learning

Academic Editor: Pingyi Fan

Received: 14 May 2026

Revised: 19 June 2026

Accepted: 23 June 2026

Published: 29 June 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

Semantic segmentation is a fundamental and important task in computer vision that divides an image into regions, where each pixel is assigned a semantic label (like “road,” “car,” or “pedestrian”) and a corresponding mask is generated to isolate each identified object or area [1]. It serves as a critical perception component for autonomous systems, including self-driving vehicles and mobile robots, which need to understand their surroundings in real time [2]. As these systems are deployed in diverse and evolving environments, the set of objects they must recognize unavoidably expands [3]. For example,

an autonomous driving vehicle trained to recognize sidewalks and pedestrians might later need to identify a new type of electric scooter or a temporary construction barrier when it enters another part of the city. Retraining the entire segmentation model from scratch every time a new class appears is time consuming, computationally expensive, and impractical for deployed systems [4]. Therefore, the ability to learn new classes incrementally is essential for the stable and long-term deployment of intelligent autonomous systems.

However, standard deep learning models like Convolutional Neural Networks (CNNs) usually lack the inherent ability to learn continually. When a model trained on a specific set of classes is updated with data from a new category, it tends to overwrite its existing parameters to minimize the loss on the new data. This problem is known as catastrophic forgetting in semantic segmentation [5,6], and it will result in a drastic performance drop on previously learned tasks. To mitigate this, researchers have proposed various Continual Learning (CL) strategies, ranging from data replay methods that store subsets of old data [4] to regularization techniques that constrain weight updates [7,8]. While effective in some scenarios, these methods often introduce significant memory overhead or fail to completely prevent forgetting in dense prediction tasks like semantic segmentation [9]. Moreover, many existing CL approaches are designed for heavy, offline models and do not work well with lightweight, real-time models like our backbone model PIDNet [10] for this research, where model size and frames per second (FPS) are the most important considerations.

To explicitly illustrate the severity of catastrophic forgetting and the efficacy of the proposed solution, a direct visual comparison is presented in Figure 1. When a standard semantic segmentation model is naively fine-tuned to learn a novel category—such as the “Bus” class—it suffers from severe parameter interference. As shown in Figure 1d, the naive approach completely loses its contextual understanding of the scene, producing noisy predictions (highlighted in red) that erroneously classify background elements, such as the billboard on the right, as the new object. In stark contrast, Figure 1c demonstrates the output of the proposed continual learning framework. By structurally isolating the learning process, the model successfully identifies the new class with sharp boundaries (highlighted in green), matching the ground truth in Figure 1b without corrupting the original spatial representations of the input image (Figure 1a).

Therefore, to bridge the gap between heavy, offline continual learning methods and real-time segmentation models, a lightweight, real-time strategy for the state-of-the-art segmentation architecture is proposed. This approach utilizes the model’s parallel branching structure to introduce plasticity for new classes while preserving stability for trained classes. The main contributions of this paper fall into three parts:

- **Core method.** We augment PIDNet, a state-of-the-art real-time segmentation model, with the ability to learn new classes after deployment, turning a fixed offline network into one that adapts continually. The key idea is knowing where to add this new capacity. Instead of spreading trainable parts throughout the network, we attach a single lightweight branch at one carefully chosen location, which is the boundary (Derivative) branch of PIDNet. This branch does not work from edges alone. It reuses the frozen network’s contextual features and uses the boundary signal to locate new objects within the scene. We achieve all of this without any knowledge distillation or teacher network, which sets our method apart from earlier approaches that modify the whole backbone (Section 2).
- **Training and inference.** Training is simple and inexpensive. We update only the new branch and its class head, using only the data of the new class, with no stored old data and no distillation. At inference, a confidence threshold decides for each pixel whether to trust the new branch or to fall back to the frozen network, so the old and new predictions stay on separate paths. Because the two are never forced into

one shared classifier, the model avoids the background shift problem that troubles methods which rely on a single combined classifier.

- **Evaluation.** We test the method thoroughly on the Cityscapes benchmark [2]. This includes a comparison against joint training and fine-tuning baselines, a comparison with leading continual segmentation methods, an ablation that pinpoints the best location for the branch, a report of efficiency (parameters, FLOPs, latency, and memory), and a study of how the routing threshold behaves across different classes. The results show that PILOT keeps its accuracy on the base classes while also learning the new ones, adding fewer than 5% extra parameters and maintaining real-time speed.

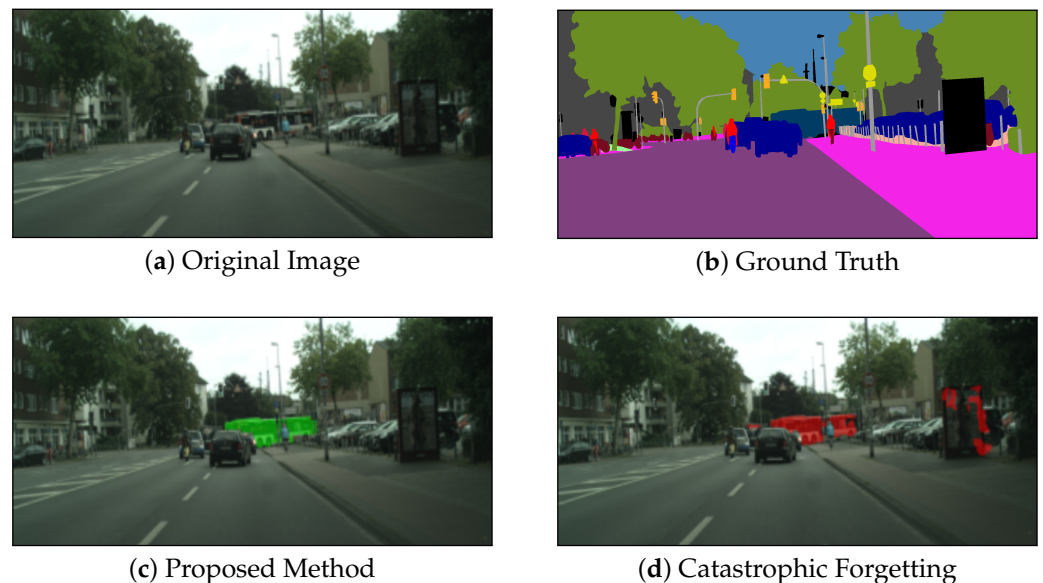


Figure 1. Visual comparison of catastrophic forgetting versus the proposed continual learning framework. (a) The original input image. (b) The ground-truth segmentation map. (c) The proposed method accurately segments the newly introduced “Bus” class (highlighted in green) without disrupting the scene. (d) Naive fine tuning results in severe catastrophic forgetting, producing noisy and inaccurate predictions (highlighted in red) that confuse the new class with background elements like the billboard.

The remainder of this paper is organized as follows. Section 2 reviews related work in real-time semantic segmentation and continual learning. Section 3 details the proposed continual learning architecture and training methodology. Section 4 presents the experimental setup, datasets, and quantitative results. Finally, Sections 5 and 6 conclude the paper and discuss future directions.

2. Related Work

Real-time semantic segmentation has received considerable attention due to its importance in autonomous driving, robotics, and embedded perception. Early lightweight models such as ENet [11] and ICNet [12] demonstrated that aggressive model compression and reduced-resolution processing can achieve high frame rates, although often with significant accuracy degradation. Subsequent approaches focused on multi-branch network designs to balance semantic context and spatial precision. For example, BiSeNet [13] introduced a dual-path architecture separating spatial detail and semantic reasoning. Expanding on this efficiency, models like Fast-SCNN [14] utilized shared shallow networks to reduce computational overhead, while STDC [15] and DDRNet [16] optimized spatial details and deep contextual features without relying on complex attention modules. More recently, PID-Net [17] proposed a three-branch structure consisting of a P-branch for semantic context, an

I-branch for intermediate feature refinement, and a D-branch dedicated to high-frequency boundary detection. This architecture provides state-of-the-art accuracy among real-time models, making it attractive for deployment in dynamic environments. The continued relevance of this architecture was recently highlighted by Tan and Jin [10], who demonstrated that modifying PIDNet's proportional and integral branches with complex multi-scale feature fusion can further enhance baseline road sensing capabilities. These efforts primarily focus on scaling the contextual branches to maximize static, offline accuracy. Consequently, the potential of the lightweight D-branch for real-time continual adaptation remains largely unexplored. Indeed, similar to other standard real-time segmentation networks, the baseline PIDNet remains strictly trained offline and lacks inherent mechanisms for continual adaptation once deployed.

Beyond balancing speed and accuracy, accurately delineating object boundaries has emerged as a critical focus in modern segmentation architectures. Frameworks such as GSCNN [18] introduced a dedicated shape stream to process spatial boundaries independently from the regular texture stream, proving that decoupling edge information significantly improves pixel-level classification along object borders. Similarly, PointRend [19] treated image segmentation as a rendering problem, selectively performing point-based predictions at ambiguous object boundaries to achieve crisp contours. These findings demonstrate that boundary-level features contain highly discriminative spatial cues. This fundamental characteristic is what makes the boundary-focused D-branch of PIDNet an ideal, lightweight anchor for learning new semantic categories.

Continual learning semantic segmentation aims to address the static nature of these deployed models by enabling them to learn new semantic categories over time without re-training from scratch or requiring access to previous datasets. This adaptability is especially critical because deployed systems frequently encounter highly dynamic and unstructured real-world environments where moving objects or novel obstacles can severely degrade the perception system's robustness, as observed in recent stereo vision frameworks [20]. A key challenge is catastrophic forgetting: when new-class data are introduced, old classes often appear as background, causing the model to overwrite their learned representations. Early pioneering works in this domain, such as ILT [21], attempted to adapt standard continual learning techniques by applying knowledge distillation to both the final output probabilities and the intermediate feature maps. Building on this, several frameworks have been proposed to specifically handle the background shift problem. MiB [9] modeled the background shift during incremental steps by modifying the cross-entropy and distillation losses to better handle missing labels for old classes. PLOP [22] extended this idea using pooled output distillation, which aligns multi-scale intermediate features between teacher and student models to better preserve old-class knowledge.

Another direction incorporates pseudo-labeling and unknown-class modeling. SSUL [23] introduced an "unknown" label to identify unlabeled old-class regions and employed saliency-based masks to avoid collapsing old objects into the background. IDEC [24] further advanced pseudo-labeling through dynamic thresholding and integrated adversarial and contrastive distillation (DADA and ARCL) to stabilize feature representations across incremental steps. These methods offer strong performance but generally rely on heavy backbones such as ResNet-101 [25] or DeepLab-style decoders [26], making them unsuitable for real-time settings.

Architectural approaches provide an alternative path. RCIL [27] introduced Representation Compensation Networks, where each convolutional block is decomposed into a frozen branch preserving old representations and a trainable branch responsible for learning new categories. Distillation is performed through pooled-cube features of the frozen teacher network. A related method, UCD [28], incorporated uncertainty-aware

contrastive distillation to emphasize reliable feature regions during knowledge transfer. While effective, these frameworks also introduce computational overhead that makes them difficult to deploy in high-FPS applications.

Despite the progress of these continual learning semantic segmentation methods, two limitations persist: (i) most are designed around high-capacity models unsuitable for embedded deployment, and (ii) their stability plasticity mechanisms do not translate well to lightweight architectures that prioritize speed. Many incorporate multi-scale feature distillation, auxiliary networks, or additional classifier heads that increase computational cost.

In contrast, this paper targets continual learning for real-time segmentation by exploiting the inherent functional decomposition of PIDNet rather than by augmenting a heavy offline backbone. The distinction from the most closely related architectural approaches is concrete rather than incremental. RCIL [27] decomposes every convolutional block into a frozen branch and a trainable compensation branch distributed throughout the network, and it still depends on pooled-cube knowledge distillation from a frozen teacher; UCD [28] likewise relies on uncertainty-aware contrastive distillation to transfer knowledge. Both therefore require a teacher forward pass and a distillation objective at every incremental step, which roughly doubles the computation per step and is tailored to offline ResNet-101 or DeepLab-style models.

Our method departs from this paradigm in three respects. First, plasticity is localized. Instead of inserting trainable branches across all blocks, a single lightweight branch is attached at the one location that the functional decomposition of the backbone identifies as most suitable, namely the Derivative branch that captures object boundaries, which we validate by ablation in Section 4.4. Second, the method uses no distillation. No teacher network is evaluated and no distillation loss is used, so the incremental step costs only a forward and backward pass through the small parallel modules. Third, the base and novel predictions are kept on separate pathways and combined by confidence routing, which structurally sidesteps the background shift problem that methods based on a unified classifier and distillation are designed to counteract. The result is a design principle, transferable to other real-time architectures with multiple branches, that injects continual plasticity at the boundary pathway and thereby attains incremental learning without replay or distillation as well as without sacrificing real-time inference, addressing a gap left by prior continual semantic segmentation frameworks.

3. Materials and Methods

3.1. Problem Formulation and Architecture Overview

In class-incremental semantic segmentation, a model is trained on a sequence of tasks where each new task introduces novel semantic categories. Let $\mathcal{C}_0$ represent the set of base classes (e.g., the original 15 classes) learned during the initial training phase using dataset $\mathcal{D}_0$. In subsequent incremental steps $t \in \{1, \dots, T\}$, the model receives a new dataset $\mathcal{D}_t$ containing images annotated only for a new set of classes $\mathcal{C}_t$. The objective of continual learning is to update the model to accurately segment all encountered classes $\mathcal{C}_{0:t} = \mathcal{C}_0 \cup \dots \cup \mathcal{C}_t$ while strictly utilizing only the current dataset $\mathcal{D}_t$ without retraining on previous data. The primary challenge in this setting is preventing the catastrophic forgetting of earlier classes $\mathcal{C}_{0:t-1}$ while maintaining the plasticity to learn $\mathcal{C}_t$.

To address this, a lightweight, parallel-branch architecture built upon the PIDNet backbone is proposed. An overview of the PILOT framework is illustrated in Figure 2. The architecture is conceptually divided into two distinct parts: a frozen base network and an unfrozen incremental framework.

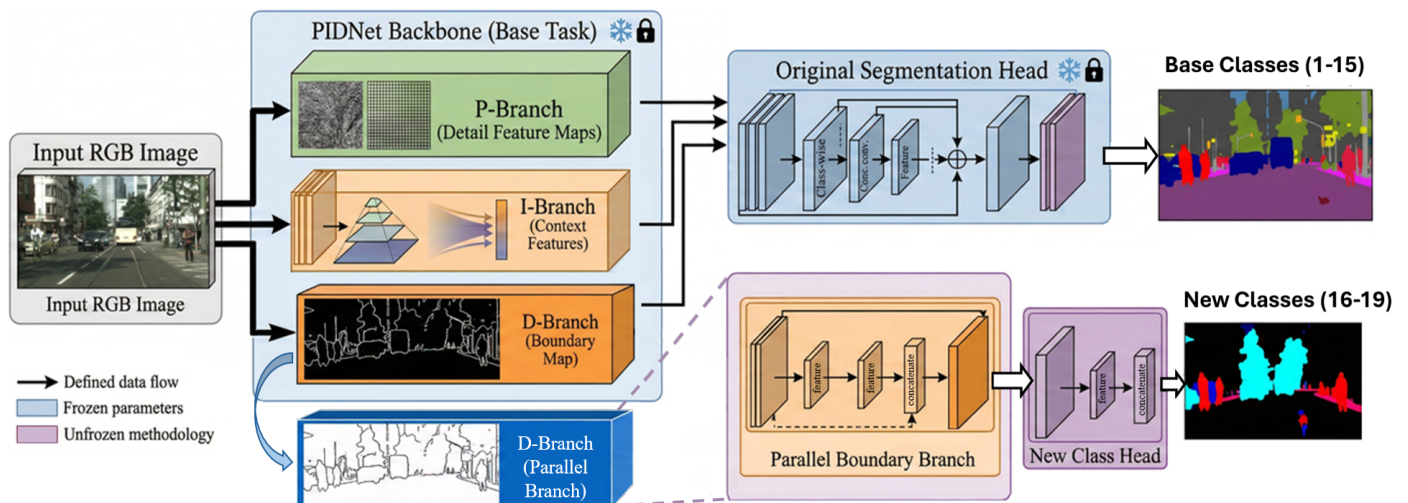


Figure 2. Overview of the proposed continual learning framework. The original PIDNet backbone and segmentation head (blue panels) are frozen after the base task to prevent catastrophic forgetting. To learn new semantic categories incrementally, an unfrozen Parallel Boundary Branch (purple panel) is introduced. This branch runs in parallel with the frozen D-branch, sharing the same input from the RGB image, and bypasses the need to retrain the contextual P- and I-branches.

The base network consists of the original PIDNet architecture, which includes the Proportional (P), Integral (I), and Derivative (D) branches, along with the original segmentation head. After the base training phase on $\mathcal{C}_0$, all parameters within this main backbone and original head are strictly frozen. This ensures that the foundational feature extraction capabilities and the semantic boundaries of the original 15 classes are permanently preserved, establishing a hard defense against catastrophic forgetting.

To accommodate novel classes, the Parallel Boundary Branch is introduced. As shown in Figure 2, this unfrozen branch runs in parallel with the frozen D-Branch, reusing the frozen backbone's contextual features through projection layers rather than relearning scene semantics. By learning high-frequency boundary representations tailored to the novel classes, the parallel network can isolate and define the contours of new objects without interfering with the primary contextual pathways (P- and I-branches). The features from this parallel branch are then passed to a dedicated New Class Head, which outputs the segmentation masks for the incremental categories.

This dual-pathway design allows the model to route base class predictions through the original, undisturbed network while dynamically learning new categories through the isolated parallel structure.

3.2. The Frozen PIDNet Backbone

The foundation of the proposed framework relies on the PIDNet architecture, which is a real-time semantic segmentation model inspired by Proportional–Integral–Derivative (PID) controllers. To efficiently balance spatial details and global context, the network utilizes a three-branch structure:

- Proportional (P) Branch: designed to process and maintain semantic context across the image.
- Integral (I) Branch: responsible for intermediate feature refinement, aggregating contextual information to support robust region classification.
- Derivative (D) Branch: dedicated exclusively to high-frequency boundary detection, extracting the sharp edges and contours of semantic objects.

In a standard deployment, the features from these three branches are fused in the final stages to produce the segmentation map. However, in the proposed continual learning framework, the entire pre-trained PIDNet backbone and its original segmentation head are strictly frozen following the initial training phase on the base classes.

Freezing the base architecture serves two critical functions. First, it guarantees that the established feature representations and boundary definitions for the original classes remain entirely uncorrupted when new data are introduced. By preventing any weight updates in the main pathways, the model is immunized against catastrophic forgetting. Second, this constraint significantly reduces the computational overhead during incremental learning steps. Since gradients are not computed or updated for the massive base network, the training process remains lightweight.

3.3. The Parallel Boundary Branch

To introduce plasticity for novel semantic categories without disrupting the frozen base network, a lightweight Parallel Boundary Branch is proposed. As established in the ablation study (Section 4.4), structurally aligning this incremental branch with the Derivative (D) branch of PIDNet yields the optimal balance between preserving base-class stability and learning new features.

The fundamental premise of this design is that high-frequency boundary information is particularly effective for delineating new semantic objects. Let $I \in \mathbb{R}^{3 \times H \times W}$ denote the input RGB image. In our framework, I is processed by two D-branches that operate in parallel: the original frozen D-branch produces a boundary feature map $F_D \in \mathbb{R}^{C \times H \times W}$ used by the original segmentation head for base-class prediction, while an unfrozen Parallel Boundary Branch produces a novel-class boundary feature map $F_{D'} \in \mathbb{R}^{C \times H \times W}$ in parallel, reusing the contextual features of the frozen backbone. This parallel design ensures that the boundary representations learned for novel classes do not constrain or modify the frozen base-class features.

The parallel branch shares the architectural design of the original D-branch, and crucially, it is fed by the same intermediate representations of the frozen backbone rather than operating on the raw image in isolation. Concretely, the branch is seeded from the shared frozen stem and is then augmented at two stages by contextual features drawn from the frozen Integral (I) branch through lightweight, trainable projection layers, which is in direct analogy to how the original D-branch is informed by the contextual trunk. Let $F_I^{(3)}$ and $F_I^{(4)}$ denote the frozen I-branch feature maps at the two intermediate stages, and let S denote the shared low-level stem features. The novel-class feature map is then formed as

$$F_{D'} = \Phi_{\text{parallel}}(S, \pi_3(F_I^{(3)}), \pi_4(F_I^{(4)}); \theta_{\text{parallel}}),$$

where $\pi_3(\cdot)$ and $\pi_4(\cdot)$ are the trainable projections that inject frozen contextual information into the parallel branch, and θ_{parallel} are the only parameters updated for the novel class. The parallel branch therefore does not attempt to recognize new objects from edges alone; rather, it learns a boundary-guided readout on top of the rich semantic context that the frozen backbone already provides.

This design directly explains why freezing the Proportional (P) and Integral (I) branches does not limit the recognition of new classes. The frozen P and I branches encode general-purpose scene context, such as the layout of the road, sidewalk, vegetation, and surrounding traffic, that was learned during base training and that transfers almost unchanged to novel categories: a bus still rests on a road surface, and a bicycle still appears beside pedestrians and sidewalks. The novel-class problem is therefore not one of re-learning scene semantics but rather of carving out the precise spatial extent of the new object within an already well-understood scene. By reusing the frozen contextual features

through the projection layers π_3 and π_4 and dedicating its trainable capacity to the high-frequency boundary residual, the parallel branch obtains sufficient semantic grounding while keeping the number of new parameters extremely small (Section 4.6). Freezing the contextual branches thus preserves transferable knowledge rather than restricting plasticity, which is consistent with the strong base-class retention reported in Section 4.1.

The novel-class boundary feature map, $F_{D'}$, is then fed into a dedicated New Class Head. This head has one background channel plus one output channel for each novel class introduced so far, where the background channel covers every pixel that does not belong to a novel class. Because classes are added one at a time, the head starts with two channels at the first step and grows by one channel at each subsequent step, so the final 19-class model uses a five-channel head that retains all four novel classes. It produces the spatial logits Y_{new} :

$$Y_{\text{new}} = \mathcal{H}_{\text{new}}(F_{D'}; \theta_{\text{head}}), \quad Y_{\text{new}} \in \mathbb{R}^{(K_t+1) \times H \times W}$$

where $\mathcal{H}_{\text{new}}$ and θ_{head} represent the new segmentation head and its parameters, and K_t is the number of novel classes introduced up to step t . Keeping an explicit background channel lets the head learn what the novel classes are not learning, which is what later allows confidence routing at inference to separate novel-class pixels from everything the frozen network already handles.

By isolating the learning process strictly within θ_{parallel} and θ_{head} , the proposed method ensures that the gradient updates triggered by the novel class data never backpropagate into the base PIDNet backbone. This structural isolation effectively mitigates catastrophic forgetting while requiring only a fraction of the computational overhead compared to retraining the entire network or utilizing heavy distillation pipelines.

3.4. Training Strategy and Loss Function

During the incremental learning phase for a new task t , the model is optimized using only novel-class data $\mathcal{D}_t$ with no base-class data revisited. Unlike traditional continual learning methods that mitigate forgetting by maintaining memory-intensive replay buffers of historical data, the proposed framework is entirely replay-free.

The complete step-by-step training protocol is summarized in Algorithm 1. This protocol strictly limits weight updates to the newly introduced modules. During the initialization phase, all parameters within the foundational PIDNet backbone (θ_{base}) and the original segmentation head ($\theta_{\text{orig_head}}$) are permanently frozen. Consequently, during backpropagation, gradients are computed and applied solely to the parameters of the Parallel Boundary Branch (θ_{parallel}) and the New Class Head (θ_{head}).

To optimize these incremental components, a standard pixel-wise softmax Cross-Entropy (CE) loss is applied over the output channels of the New Class Head. For each image in the incremental dataset, we build a target Y_{seg} in which each pixel of a novel class is labeled with that class, pixels that belong to the already-known base classes are labeled as background and act as the negative class, and void or unlabeled pixels are ignored. Let Ω be the set of valid (non-ignored) pixels, let y_i be the target at pixel i , and let $\hat{y}_{i,c}$ be the softmax probability of channel c from the New Class Head. The incremental loss is

$$\mathcal{L}_{\text{inc}} = -\frac{1}{|\Omega|} \sum_{i \in \Omega} \sum_{c=0}^{K_t} \mathbb{I}(y_i = c) \log(\hat{y}_{i,c}),$$

where channel 0 is the background, channels 1 to K_t are the novel classes, and $\mathbb{I}(\cdot)$ is the indicator function. When the first class is added, this reduces to a two-channel, background-versus-novel objective.

Algorithm 1 Incremental Training via the Parallel Boundary Branch

Require: Pre-trained PIDNet backbone $\mathcal{B}$ with parameters θ_{base}
Require: Pre-trained original head $\mathcal{H}_{\text{orig}}$ with parameters $\theta_{\text{orig_head}}$
Require: Incremental dataset $\mathcal{D}_t$ containing images X and labels Y for novel classes $\mathcal{C}_t$
Require: Learning rate α , total epochs E

- 1: **Phase 1: Initialization**
- 2: Freeze backbone parameters: $\theta_{\text{base}} \leftarrow \text{requires_grad}(\text{False})$
- 3: Freeze original head: $\theta_{\text{orig_head}} \leftarrow \text{requires_grad}(\text{False})$
- 4: Initialize Parallel Boundary Branch parameters: θ_{parallel}
- 5: Initialize New Class Head parameters: θ_{head}
- 6: **Phase 2: Incremental Optimization**
- 7: **for** epoch = 1 **to** E **do**
- 8: **for** each mini-batch $(X, Y) \in \mathcal{D}_t$ **do**
- 9: 1. *Forward pass through unfrozen parallel branch*
- 10: Generate novel-class boundary features: $F_{D'} \leftarrow \Phi_{\text{parallel}}(X; \theta_{\text{parallel}})$
- 11: Generate novel class predictions: $\hat{Y}_{\text{new}} \leftarrow \mathcal{H}_{\text{new}}(F_{D'}; \theta_{\text{head}})$
- 12: 2. *Loss computation (novel classes vs. background)*
- 13: Compute softmax Cross-Entropy Loss: $\mathcal{L}_{\text{inc}} = \text{CE}(\hat{Y}_{\text{new}}, Y_{\text{seg}})$
- 14: 3. *Backpropagation and targeted parameter update*
- 15: Compute gradients: $\nabla \theta_{\text{parallel}}, \nabla \theta_{\text{head}} \leftarrow \frac{\partial \mathcal{L}_{\text{inc}}}{\partial \theta_{\text{parallel}}}, \frac{\partial \mathcal{L}_{\text{inc}}}{\partial \theta_{\text{head}}}$
- 16: Update parallel branch: $\theta_{\text{parallel}} \leftarrow \theta_{\text{parallel}} - \alpha \nabla \theta_{\text{parallel}}$
- 17: Update new head: $\theta_{\text{head}} \leftarrow \theta_{\text{head}} - \alpha \nabla \theta_{\text{head}}$
- 18: **end for**
- 19: **end for**
- 20: **return** Updated incremental parameters θ_{parallel} and θ_{head}

Note that the negative class is defined from the base-class pixels that are present in the incremental images, not from the novel class alone, so the head is supervised to keep these regions on the background channel. The base-class identities themselves are never relearned, because the original 15-class labels are absent from $\mathcal{D}_t$ and the frozen backbone and original head are not updated. By restricting the loss computation and weight updates exclusively to the lightweight parallel modules, the training process avoids the massive computational burden associated with full-network fine tuning or complex multi-scale feature distillation pipelines. This targeted optimization ensures rapid, resource-efficient adaptation to novel semantic categories, maintaining the low inference latency required for real-time deployment.

To generate the final segmentation map during deployment, a confidence-based routing mechanism is utilized to seamlessly merge the predictions from the unfrozen parallel branch and the frozen base network. The complete inference data flow is illustrated in Figure 3.

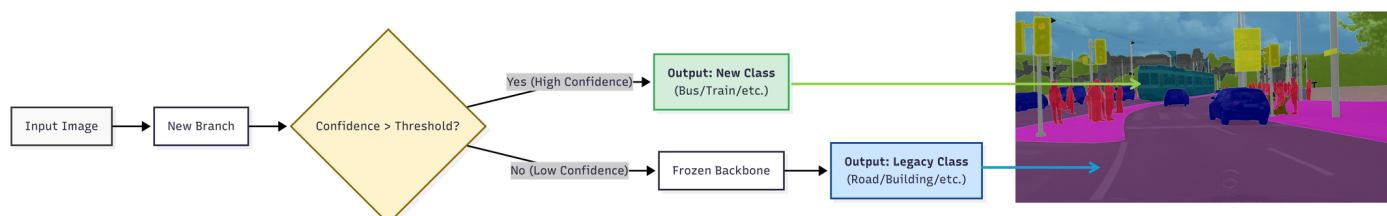


Figure 3. The inference prediction routing mechanism. The model first evaluates the input image through the new parallel branch. Pixels exceeding the confidence threshold are classified as the new category. For low confidence regions, the model falls back to the frozen backbone to accurately predict legacy classes, effectively merging the two pathways into a single, coherent segmentation map.

When an input image is processed, the Parallel Boundary Branch first evaluates the scene to detect any newly learned semantic categories. The dedicated New Class Head outputs a probability map for these novel objects. A predetermined confidence threshold (τ) is then applied to these predictions. If the confidence score for a specific pixel exceeds this threshold, the model assigns it to the novel class (e.g., Bus or Train).

Conversely, if the confidence score falls below the threshold, the system defaults to the foundational knowledge of the frozen PIDNet backbone. The frozen network then processes those low confidence regions to assign legacy classes (e.g., Road or Building). This conditional routing strategy ensures that novel objects are only segmented when the model is highly certain. It strictly preserves the integrity of the base classes in all other areas. By intelligently routing the predictions rather than aggressively overwriting them, the framework maintains the high accuracy and real-time efficiency required for autonomous perception.

4. Results

We evaluate the PILOT framework on the Cityscapes dataset [2], which is a large-scale benchmark for urban scene understanding. Cityscapes contains 5000 finely annotated images collected from 50 cities across Germany and neighboring countries, which are split into 2975 training images, 500 validation images, and 1525 test images. Each image has a resolution of 2048×1024 pixels and is annotated with pixel-level labels across 19 semantic classes, including road, sidewalk, building, vegetation, person, car, bus, train, motorcycle, and bicycle, among others. Following standard continual semantic segmentation protocols, we use the 2975 training images for both base training and incremental learning steps, and we report all mIoU scores on the 500 validation images.

The following subsections represent a comprehensive evaluation of the PILOT framework on the Cityscapes dataset under a class-incremental semantic segmentation protocol. We begin in Section 4.1 by analyzing the long-term continual learning behavior of PILOT across all three PIDNet variants (S, M, and L), comparing against Joint Training (upper bound) and Fine-Tuning (lower bound) baselines under a matched-backbone protocol. We then position PILOT against state-of-the-art continual semantic segmentation methods on the 14-1 and 10-1 benchmarks in Section 4.2. Section 4.3 examines the stability–plasticity trade-off through per-step mIoU trajectories. Section 4.4 provides an ablation analysis of the proposed parallel branch connection point, and Section 4.5 examines the sensitivity of the model’s performance to the decision threshold of the new class head. Finally, Section 4.7 presents qualitative segmentation results visualizing the model’s incremental learning behavior across model capacities.

4.1. Continual Learning

We evaluate the long-term continual learning capability of our method by extending the experiment from the base 15 classes to the full 19 classes, adding one class at a time across four incremental steps. The quantitative results at each incremental stage are detailed in Table 1. To contextualize the performance of our method, we establish reference baselines using Joint Training and Fine Tuning, which were both implemented on the PIDNet-M backbone for fair comparison.

Joint Training (Upper Bound). This setting represents the ideal offline scenario in which the model is retrained from scratch on the full accumulated dataset whenever a new class is introduced. While this yields strong performance (maintaining approximately 78% mIoU throughout), it requires access to all historical data at every step and incurs substantial retraining cost. Joint Training is therefore impractical for real-world autonomous driving deployment and serves only as a theoretical upper bound.

Table 1. Quantitative analysis of incremental learning (Steps 15 to 19). Part I reports the Overall mIoU (all classes known at that step). Part II reports the Base Class Stability (mIoU calculated only on the original 15 classes), showing how well the model resists forgetting. Best results are shown in bold.

Method	Backbone	Base Task		Incremental Steps		
		Step 0 (15 cls)	Step 1 (16 cls)	Step 2 (17 cls)	Step 3 (18 cls)	Step 4 (19 cls)
Joint Training	PIDNet-M	80.92	78.90	78.50	78.10	77.80
Fine Tuning	PIDNet-M	80.92	70.08	65.12	61.75	59.90
Part I: Overall Performance (mIoU on All Classes)						
New-head-only	PIDNet-M	80.92	68.54	65.09	63.61	62.35
Partial D-FT	PIDNet-M	80.92	68.44	65.74	65.13	65.57
Full D-FT	PIDNet-M	80.92	68.77	66.13	65.23	65.08
PILOT	PIDNet-S	79.02	72.71	69.94	68.25	68.91
PILOT	PIDNet-M	80.92	74.80	72.03	71.71	71.87
PILOT	PIDNet-L	80.58	75.00	72.03	70.97	70.54
Part II: Base Class Stability (mIoU on Original 15 Classes Only)						
New-head-only	PIDNet-M	80.92	71.11	70.89	70.95	70.74
Partial D-FT	PIDNet-M	80.92	70.02	70.01	70.10	69.73
Full D-FT	PIDNet-M	80.92	70.24	70.13	70.17	69.88
PILOT	PIDNet-S	79.02	77.24	75.32	75.21	75.19
PILOT	PIDNet-M	80.92	79.14	77.37	77.90	77.38
PILOT	PIDNet-L	80.58	79.31	77.32	77.29	76.76

Fine Tuning (Lower Bound). This setting represents a naive incremental approach in which the model is sequentially fine-tuned on new class data without access to previous samples or knowledge-preservation mechanisms. As shown in Table 1, this approach suffers from progressive performance degradation, dropping from 80.92% to 59.90% over four incremental steps. This confirms that without an explicit mechanism for preserving prior knowledge, incremental training is inadequate for long-term class addition.

Matched-Backbone Baselines (Intermediate References). To isolate what the Parallel Boundary Branch contributes beyond the Joint Training and Fine-Tuning references, we add three further baselines on the same frozen PIDNet-M backbone, which are all run under the identical incremental protocol and routing threshold. The New Head Only baseline trains only the New Class Head on top of the frozen original Derivative branch with no dedicated branch added. Partial D-branch fine tuning (Partial D-FT) also unfreezes the deeper layers of the original Derivative branch, and Full D-branch fine-tuning (Full D-FT) unfreezes the whole original Derivative branch. The results are reported alongside PILOT in Table 1.

All three baselines fall between the naive Fine Tuning lower bound and PILOT. New Head Only has the least trainable capacity, and its overall mIoU declines steadily as classes accumulate, reaching 62.35% at 19 classes. Fine tuning the shared Derivative branch recovers some of this, reaching about 65%, because the extra capacity helps fit the new classes. Both fine-tuning variants, however, lower base-class stability to roughly 70%, since adapting the branch that the frozen 15-class head also consumes perturbs the original predictions. New Head Only keeps the base path frozen, yet its base-class stability still falls to about 71%, because a head built on the generic frozen features is less precise and routes some base-class pixels to the novel classes. PILOT avoids both failure modes. Its dedicated parallel branch leaves the original Derivative branch untouched and produces a more discriminative head that rarely overrides base pixels, so PILOT keeps the highest base-class stability at 77.38% while also reaching the best overall mIoU at 71.87%. The comparison shows that the gain does not come from simply adding trainable parameters, because Full D-branch fine tuning adapts more of the network yet still trails PILOT on both

axes. It comes from isolating the new capacity in a parallel branch so that learning new classes does not disturb the frozen backbone.

Efficacy of Our Approach. PILOT demonstrates robust stability across all sequential incremental steps. Utilizing the PIDNet-M backbone, which demonstrates the optimal balance between capacity and plasticity in our experiments, we observe a consistent performance trajectory. At the first incremental step (16 classes), the model achieves an overall mIoU of 74.80%, effectively retaining base knowledge with a base-class stability of 79.14%. This stability persists over the long term: even after four sequential additions (19 classes), the model maintains an overall mIoU of 71.87%, with base-class mIoU at 77.38%, which is a retention of 95.6% relative to the initial 80.92% base performance. This indicates that our D-branch connection successfully mitigates catastrophic forgetting while allowing the model to accommodate novel classes. Furthermore, the PIDNet-M variant yields superior performance compared to both the S and L backbones. At the final step, PIDNet-M outperforms the Fine-Tuning baseline by 11.97 percentage points.

4.2. Comparison with State-of-the-Art Methods

We compare our method against representative continual semantic segmentation (CSS) methods on Cityscapes under two standard protocols: 14-1 (6 tasks) and 10-1 (10 tasks). Table 2 reports the mean IoU over the initial (base) classes, the novel (incremental) classes, and all classes at the final incremental step.

Before discussing the results, we briefly explain the CSS evaluation protocol. The notation “ B - N ” denotes an experimental setup in which B (Base) classes are jointly learned during the base training phase and N (New) classes are added one at a time during incremental learning. Under the 14-1 protocol, the model is first trained on 14 base classes, after which the remaining five classes (15 through 19) are introduced sequentially across five incremental steps, yielding six total tasks. Similarly, the 10-1 protocol trains on 10 base classes and adds the remaining nine classes one at a time for 10 total tasks. All of the mIoU values reported in Table 2 are measured on the final model after all incremental steps are complete but evaluated over different class subsets. Columns labeled “1–14” and “1–10” report the mIoU computed only over the base classes, measuring how well each method resists catastrophic forgetting. Columns labeled “15–19” and “11–19” report the mIoU computed only over the novel (incrementally added) classes, measuring how well each method learns new categories. The columns labeled “all” report the mIoU computed over all 19 classes, indicating overall segmentation performance.

Table 2. Continual semantic segmentation results on Cityscapes in mIoU (%). Tasks defined as C^1 – C^T (T tasks). Best results are shown in bold.

Method	14-1 (6 Tasks)			10-1 (10 Tasks)		
	1–14	15–19	All	1–10	11–19	All
PLOP [22]	63.54	15.38	48.33	60.75	27.97	42.96
MiB [9]	66.37	14.36	50.05	61.80	32.97	45.73
MiB + AWT [29]	65.60	19.19	50.72	60.97	35.70	46.55
DKD [30]	68.83	14.70	51.86	66.77	34.52	48.92
TOPICS [31]	73.03	42.47	61.74	71.37	52.62	59.36
PILOT (Ours)	76.52	49.20	70.21	75.84	54.20	65.68

It is important to note that prior CSS methods including PLOP [22], MiB [9], MiB + AWT [29], DKD [30], and TOPICS [31] adopt DeepLabV3 with a ResNet-101 backbone, whereas our method uses PIDNet for its real-time efficiency in autonomous driving deployment. While this backbone difference precludes strictly equivalent comparison, two

observations indicate the effectiveness of our continual learning framework. First, our method achieves substantially stronger base-class retention: on the 14-1 protocol, we obtain 76.52% on classes 1–14 compared to TOPICS' 73.03% and PLOP's 63.54%. Second, our method demonstrates markedly improved plasticity on novel classes, achieving 49.20% on classes 15–19 compared to 42.47% for TOPICS, 19.19% for MiB + AWT, and 15.38% or lower for the remaining baselines. The gain is consistent across both the 14-1 and 10-1 protocols.

We also want to be clear about the scope of this comparison. We do not present PILOT as a general method that can be attached to any segmentation backbone. It is designed specifically for PIDNet, and its effectiveness depends on the dedicated Derivative branch that supplies the high-frequency boundary information the parallel branch reuses to locate new classes. Heavier backbones such as DeepLabV3 with ResNet-101 do not contain a separate boundary pathway of this kind, so attaching a Parallel Boundary Branch to them would not reproduce the same mechanism and would not form a meaningful matched-backbone test. Reimplementing the prior methods on PIDNet faces the opposite problem, because those methods rely on distillation and replay procedures tuned for their original heavy backbones, which do not transfer cleanly to a lightweight three-branch network. We therefore report the cross-backbone numbers in Table 2 as evidence that our design is effective within its intended real-time setting rather than as a claim of superiority under identical conditions. The matched-backbone results in Table 1 instead isolate the contribution of the continual learning strategy on a single fixed backbone, and whether the boundary-guided design transfers to other architectures is left as an explicit direction for future work.

To further visualize the dynamics of catastrophic forgetting across methods, we plot per-step overall mIoU on the 10-1 protocol in Figure 4. The 10-1 setting is particularly informative because it requires nine sequential incremental steps, providing a clearer view of long-term stability than shorter protocols. As shown in the figure, all baseline methods exhibit a pronounced downward trajectory as more classes are added with PLOP, MiB, and MiB + AWT degrading sharply after Step 6 and ending below 50% mIoU. TOPICS maintains the strongest performance among prior approaches but still declines substantially in later steps, falling from approximately 69% at Step 5 to 59.36% at Step 9. In contrast, our method maintains consistently higher mIoU throughout the entire incremental sequence, ending at 65.68% at Step 9, which is a margin of 6.32 percentage points over TOPICS. Notably, our curve also exhibits a flatter trajectory, indicating that the per-step performance loss from each incremental step is smaller than for the baselines. This stability is consistent with our base-class retention results in Table 2 and supports our claim that the proposed framework more effectively mitigates catastrophic forgetting under long incremental sequences.

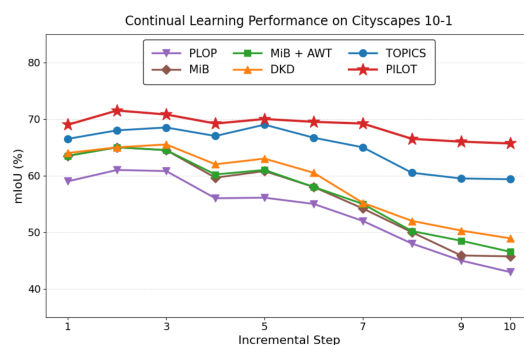


Figure 4. Per-step overall mIoU on the Cityscapes 10-1 protocol across nine sequential incremental steps. Our method (red) maintains consistently higher and more stable performance than all prior CSS approaches throughout the entire incremental sequence. Baseline curves were digitized from Hindel et al. [31] (Figure 5).

These results, together with the matched-backbone ablation in Table 1, suggest that our framework addresses both the stability–plasticity trade-off and the semantic background shift problem more effectively than prior approaches on Cityscapes continual segmentation benchmarks.

4.3. Dynamic Stability Analysis

To further investigate the stability–plasticity trade-off, we analyze the mIoU evolution across sequential incremental steps in Figure 5.

As shown in the plot, all three model variants experience a noticeable drop in performance when the first new class is introduced (from 15 to 16 classes). This initial decline is expected as the model transitions from a closed-set training regime to incremental learning. Several factors contribute to this drop. First, the Parallel Boundary Branch and new class head are randomly initialized and require training to produce reliable predictions for the novel class. Second, the confidence-based routing mechanism may occasionally misclassify base-class pixels as the novel category before the new head is fully calibrated. Third, the mIoU computation itself changes scale as additional classes are introduced into the evaluation. Notably, our framework largely avoids the classical semantic background shift problem that plagues unified-classifier CSS methods, since the frozen original head and the New Class Head operate on separate prediction pathways.

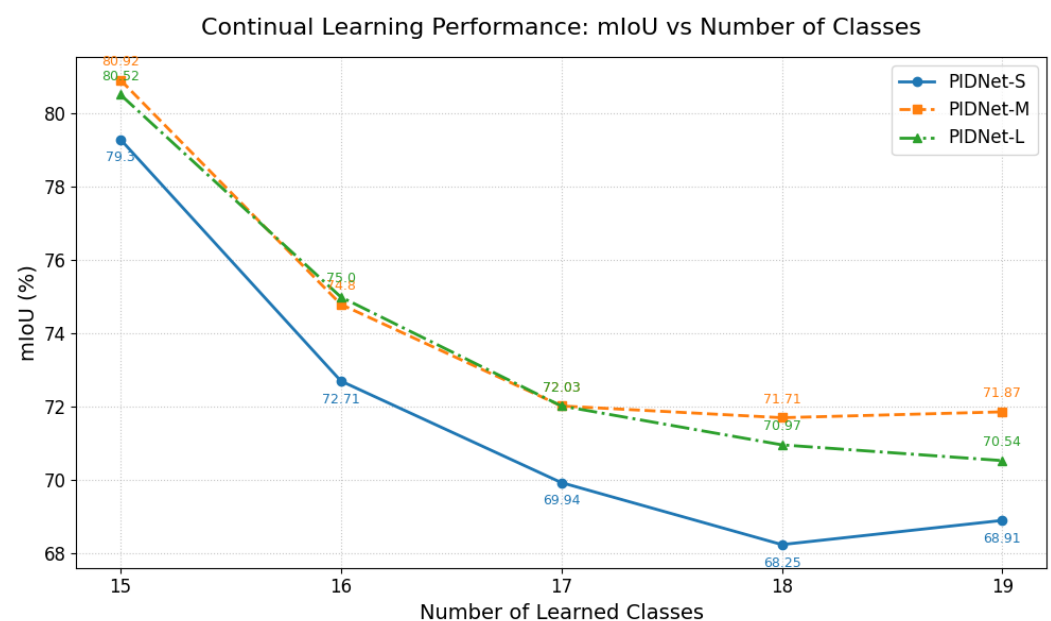


Figure 5. Continual learning performance (mIoU vs. number of classes). The plot illustrates the performance trajectory of PILOT with PIDNet-S, M, and L backbone variants across incremental steps bounded by the Joint Training upper bound and Fine-Tuning lower bound. PIDNet-M (dashed line) demonstrates the highest robustness among our variants, stabilizing above 71% at the final step.

Critically, performance stabilizes significantly after this first step. From 17 classes onward, the decline becomes substantially slower and the curve flattens. This indicates that our D-branch connection effectively limits catastrophic forgetting once the model has adjusted to the incremental setting. By isolating boundary features, the network accommodates new classes without aggressively overwriting the feature representations of the original 15 categories.

Regarding model capacity, PIDNet-M consistently achieves higher mIoU than both PIDNet-S and PIDNet-L across incremental steps. This behavior is explained by the size of the trainable incremental branch reported in Section 4.6. Because the base network is frozen, plasticity is governed entirely by the parameters of the Parallel Boundary Branch,

and these scale with backbone width: the branch contains 0.30 M trainable parameters for PIDNet-S, 1.06 M for PIDNet-M, and 1.77 M for PIDNet-L. The PIDNet-L branch is therefore 67% larger than that of PIDNet-M, yet it is optimized on exactly the same limited single-class data at each step. This larger trainable capacity makes PIDNet-L more prone to overfitting the few annotated pixels of the novel class, which degrades its generalization on the validation set. PIDNet-S, in contrast, has too few trainable parameters to fully capture the boundary characteristics of new objects. PIDNet-M thus occupies a favorable middle ground: enough capacity to learn discriminative boundary features for the new class but not so much that it overfits the scarce incremental data. The small absolute spread among the three variants at the final step (PIDNet-M: 71.87%, PIDNet-L: 70.54%) nonetheless indicates that the framework is robust across model capacities.

4.4. Ablation Analysis

PIDNet [17] decomposes feature learning into three parallel branches: the Proportional (P) branch, which preserves high-resolution detail; the Integral (I) branch, which aggregates contextual information through low-resolution processing; and the Derivative (D) branch, which extracts high-frequency boundary features to guide segmentation. To determine the optimal injection point for the parallel incremental branch, we conducted an ablation study by attaching the branch to each of these three branches of the frozen PIDNet backbone. The quantitative results are summarized in Table 3.

All three connection points preserve base-class accuracy within a narrow range (P: 76.75%, I: 77.49%, D: 77.22%), which is only a modest decrease from the 79.02% baseline, which shows that the choice of attachment point has little effect on stability. The decisive difference lies in plasticity and the ability to learn the new class: attaching the branch to the D-branch reaches a final overall mIoU of 72.65% on the 16-class task, which is clearly ahead of the I-branch (70.25%) and the P-branch (67.97%). The D-branch is therefore the optimal connection point, and the reason for this ordering, in particular why the boundary pathway suits incremental learning better than the contextual P and I branches, is analyzed next.

Table 3. Ablation analysis of connection points. The D-branch (bold) achieves the best trade-off between stability (Base mIoU) and plasticity (Overall mIoU).

Connection Point	Baseline	Incremental Step (After Adding Class)	
	Original mIoU (15 Classes)	Base mIoU (Old Classes)	Overall mIoU (All 16 Classes)
P Branch	79.02%	76.75%	67.97%
I Branch	79.02%	77.49%	70.25%
D Branch	79.02%	77.22%	72.65%

Optimal injection point. The ordering in Table 3 (D > I > P) follows directly from what each branch represents and from the constraints of the incremental setting. The Proportional (P) branch preserves high-resolution appearance and texture but carries comparatively weak contextual aggregation; attaching the new readout there forces it to infer a novel object’s identity largely from local texture, which is precisely where new classes are most easily confused with visually similar base classes (for example, a bus with a truck, or a bicycle with a pedestrian). This is why the P-branch yields the weakest plasticity. The Integral (I) branch supplies strong semantic context but at low spatial resolution, so a readout placed there localizes the often thin or elongated novel objects poorly, giving an intermediate result. The Derivative (D) branch occupies the most favorable position because, as detailed in Section 3, it is not a context-free edge detector: it combines the frozen contextual features injected from the I-branch with a high-frequency boundary residual

evaluated at the boundary-head resolution. It therefore inherits the semantic grounding of the I-branch while retaining the spatial precision needed to delineate new-object contours.

A second factor is the resistance of the boundary pathway to overfitting under scarce supervision. During each incremental step, the branch is trained on the limited pixels of a single novel class, a regime in which models with more free parameters and denser activations are prone to overfitting. The D-branch path is narrow, and its informative activations are spatially concentrated along object contours rather than spread densely across every region, so it exposes fewer effective degrees of freedom to the optimizer than a full-resolution appearance pathway would. This makes it comparatively robust to the small per-class sample sizes that characterize continual segmentation, and it is consistent with two other observations in this paper: the very small parameter footprint of the incremental modules (Section 4.6) and the finding that the more constrained PIDNet-M backbone generalizes better than the higher-capacity PIDNet-L (Section 4.3). Boundary-guided features are thus especially well suited to the incremental classes considered here (Bus, Train, Motorcycle, and Bicycle), which are all foreground “thing” classes whose silhouettes are highly distinctive against the shared urban background even when their texture overlaps with that of base classes, which is a property that is visually evident in the bicycle-versus-pedestrian separation shown in Section 4.7.

4.5. Hyperparameter Analysis

To maximize the segmentation performance of the newly introduced classes, the decision threshold (τ) for the Parallel Boundary Branch was systematically evaluated. Because the New Class Head outputs continuous probability maps, selecting an optimal τ is critical for balancing false positives and false negatives during inference.

The performance metrics for the novel class, including Intersection over Union (IoU), Precision, and Recall, were analyzed across a range of threshold values. As illustrated in Figure 6, increasing the threshold naturally improves precision by filtering out low-confidence predictions, but it simultaneously degrades recall as more true-positive pixels are rejected (the precision and recall at the operating point are reported per class in Table 4). Based on this analysis, we fix a single global operating point of $\tau = 0.75$ on the Cityscapes validation set and apply it unchanged to every novel class and every incremental step without any per-class or per-step tuning. A relatively high value is preferred so that the confidence routing stays conservative and the New Class Head overrides the frozen backbone only when it is highly confident, which protects the base classes. The per-class analysis below confirms that this choice is not delicate, so the reported results do not hinge on the exact value of τ .

The threshold-independent discriminability of the New Class Head is reported per class via the Area Under the ROC Curve (AUC) in Table 4 below. Having fixed $\tau = 0.75$ as the global operating point, we next examine whether this single threshold is appropriate across the four novel classes, which differ considerably in object scale and appearance.

Having fixed a single global decision threshold, a natural concern is whether one value of τ generalizes across semantically and geometrically diverse novel classes. To examine this, we analyze the decision threshold for all four incrementally added classes (Bus, Train, Motorcycle, and Bicycle) on the final 19-class model. For each class, we sweep τ over the full range, and we record the threshold-independent discriminability via the Area Under the ROC Curve (AUC), the per-class optimal threshold τ^* , and the IoU obtained both at τ^* and at the single global value $\tau = 0.75$ used to produce all of our main results. All metrics are computed only over benchmark pixels (void and ignore regions are excluded, which is consistent with the standard Cityscapes evaluation protocol). The results are reported in Table 4.

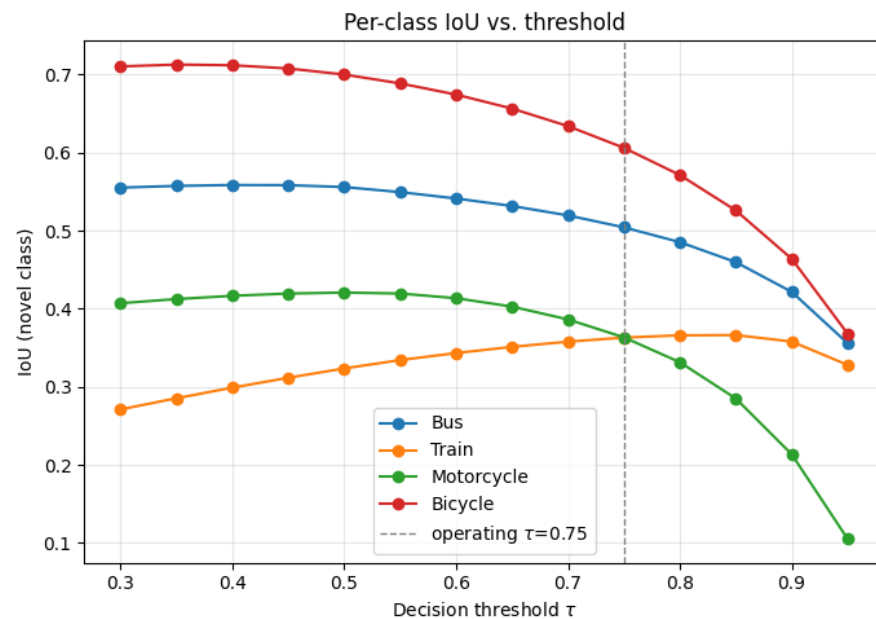


Figure 6. Per-class Intersection over Union (IoU) of the New Class Head as a function of the decision threshold τ for the four incrementally added classes on the final 19-class model; the dashed line marks the single global operating point $\tau = 0.75$ used for all main results. The IoU stays high over a wide range of τ for every class and the per-class optima differ only mildly (Table 4), showing that one global threshold is a reasonable compromise. All curves are computed over benchmark pixels only (void and ignore regions excluded).

Table 4. Per-class threshold robustness of the New Class Head on the final 19-class model. “ τ^* ” is the per-class optimal threshold; “IoU@ τ^* ” and “IoU@0.75” are the IoU at the per-class optimum and at the single global threshold used for all main results, respectively. The threshold-independent AUC exceeds 0.99 for every class, confirming that the New Class Head is strongly discriminative regardless of the chosen operating point. The single global $\tau = 0.75$ remains within a few IoU points of the per-class optimum for most classes.

Novel Class	τ^*	IoU@ τ^*	IoU@0.75	Prec@0.75	Rec@0.75	AUC
Bus	0.40	0.558	0.504	0.849	0.554	0.995
Train	0.85	0.366	0.363	0.525	0.540	0.990
Motorcycle	0.50	0.421	0.363	0.797	0.400	0.994
Bicycle	0.35	0.713	0.606	0.947	0.627	0.997
Mean	—	0.514	0.459	0.780	0.530	0.994

Two observations follow from Table 4. First, the New Class Head is highly discriminative for every novel class: the AUC ranges from 0.990 (Train) to 0.997 (Bicycle) with a mean of 0.994. Because the AUC is computed by integrating over all thresholds, this metric is independent of the particular operating point and therefore cannot be inflated by tuning τ . The consistently high AUC indicates that the separability of novel-class pixels is an intrinsic property of the learned boundary features rather than an artifact of threshold selection, which directly addresses the concern that our reported performance might depend on a cherry-picked threshold. Second, the per-class optimal thresholds do vary, from 0.35 for Bicycle to 0.85 for Train, reflecting differences in object scale, rigidity, and confusability with base classes (Train, a large and visually rare class, is more conservatively gated). Nonetheless, the single global $\tau = 0.75$ used throughout our main experiments remains close to the per-class optimum for three of the four classes, and the residual gap (a mean IoU difference of 0.514 versus 0.459) quantifies the modest cost of using one fixed threshold instead of a per-class oracle. We deliberately report all main-table results under this single

fixed τ , without any per-class tuning, so that the headline numbers do not benefit from oracle threshold selection.

Routing reliability. The confidence-based routing mechanism described in Section 3 only overrides the frozen backbone when the New Class Head is confident, which is intended to preserve the integrity of the base classes. We quantify this behavior directly on the final 19-class model. Across all benchmark pixels, the rate at which a correctly classifiable base pixel is mis-routed to a novel class (a base-override event) is only 0.09%, confirming that the routing mechanism almost never corrupts the predictions of the frozen network. Out of all of the pixels that the mechanism does route to a novel class, 85.6% are genuinely novel-class pixels (routing precision). Conflicts, defined as pixels where both the base network and the New Class Head are simultaneously confident, are rare, occurring at only 0.41% of benchmark pixels; and when such a conflict does arise, the contested pixel truly belongs to a novel class in 82.7% of cases, so resolving the conflict in favor of the New Class Head is the correct decision the large majority of the time. Together, these statistics provide quantitative evidence that the routing strategy adds new-class plasticity while leaving the stability of the base classes essentially intact, corroborating the catastrophic-forgetting resistance reported in Section 4.1.

Robustness on low-frequency classes. Among the four incrementally added categories, Train and Motorcycle are among the least frequent classes in Cityscapes, both in the number of annotated instances and in pixel coverage, which raises the concern that a branch trained only on novel-class data might overfit when supervision is scarce. The per-class results in Table 4 indicate that this is not the dominant failure mode. The defining signature of overfitting is a strong fit on the training data accompanied by poor generalization to unseen data; here, we observe the opposite on the held-out validation set. The threshold-independent AUC of the New Class Head remains very high precisely for the two rarest classes (0.990 for Train and 0.994 for Motorcycle), meaning that on validation images, the head still ranks novel-class pixels above non-novel pixels almost perfectly. A branch that had memorized the few training instances would not produce such well-separated, calibrated confidence maps on unseen scenes. Moreover, prediction precision does not collapse for these classes (0.797 for Motorcycle and 0.525 for Train at the deployment threshold); the limiting factor is instead recall, which is consistent with the small spatial extent and low instance count of these objects—that is, with data scarcity and object scale rather than with overfitting-induced instability.

Two design factors explain this robustness. First, as described in Section 3, the Parallel Boundary Branch is a narrow, boundary-focused module whose informative activations are spatially concentrated along object contours and which reuses the frozen contextual features rather than re-learning scene semantics; it therefore exposes few effective degrees of freedom to the optimizer (only 0.30M–1.77M trainable parameters, Section 4.6), which acts as a strong implicit regularizer under limited supervision. Second, this behavior is consistent with the capacity analysis in Section 4.3, where the higher-capacity PIDNet-L generalizes slightly worse than the more constrained PIDNet-M precisely because additional trainable capacity is more prone to overfitting the scarce novel-class data; PILOT's deliberately small branch sits on the favorable side of this trade-off. Finally, adding these low-frequency classes does not destabilize previously learned categories: base-class mIoU is preserved across the incremental steps in which Train and Motorcycle are introduced (Table 1). Taken together, these observations indicate that the framework remains stable on low-frequency classes with the residual performance gap on Train attributable to genuine data scarcity rather than to overfitting; narrowing this gap for extremely rare categories, for example through lightweight class-balanced or few-shot-aware objectives, is a natural direction for future work.

4.6. Computational Efficiency

A central claim of this paper is that PILOT preserves the lightweight, real-time character of the underlying PIDNet backbone. To verify this quantitatively, we measure the computational footprint introduced by the Parallel Boundary Branch and New Class Head relative to the frozen base network, and we compare it against the DeepLabV3 with ResNet-101 backbone used by the continual learning methods in Table 2. All metrics are reported for the three PIDNet variants (S, M, and L) in the deployment configuration. Parameter counts and floating point operations (FLOPs) are computed analytically. Inference speed and latency are measured on a single NVIDIA A100 GPU (NVIDIA Corporation, Santa Clara, CA, USA) at the full Cityscapes resolution of 2048×1024 with a batch size of one, using synchronized GPU timing with warm-up over an auto-calibrated number of iterations, following the standard PIDNet speed test protocol [17]. All experiments were conducted in Google Colaboratory (Google LLC, Mountain View, CA, USA) using PyTorch 2.11.0, CUDA 12.8, and Python 3.12.13. The results are summarized in Table 5.

Table 5. Computational footprint and inference efficiency at 2048×1024 with a batch size of one, all measured in one Google Colaboratory session on a single NVIDIA A100 with the same speed test routine so the numbers reflect the networks rather than the hardware. The Added columns report the overhead of the Parallel Boundary Branch and New Class Head relative to the frozen base PIDNet and do not apply to the baseline, which is marked with a dash. DeepLabV3 with ResNet-101 is the inference backbone shared by the continual learning methods in Table 2, which is listed in the table note below. Because the inference cost is set by the network and not the training method, a single row represents that family. FPS (Base) is the frozen backbone and FPS (Deployed) is the model actually run at inference.

Model	Total (M)	Added (M)	Added (%)	GFLOPs	FPS (Base)	FPS (Deployed)	Mem (MB)
PILOT (PIDNet-S)	7.93	0.30	4.0	47.5	99.9	91.5	299
PILOT (PIDNet-M)	29.60	1.06	3.7	177.6	77.9	71.9	488
PILOT (PIDNet-L)	38.71	1.77	4.8	275.3	62.4	57.1	589
DeepLabV3 + ResNet-101 *	58.63	-	-	1933.4	-	13.4	1146

* Inference backbone shared by PLOP, MiB, MiB + AWT, DKD, and TOPICS.

Lightweight overhead. As reported in Table 5, the incremental modules are very compact. Across all three backbones, the Parallel Boundary Branch and New Class Head add only 0.30 M to 1.77 M parameters, which is an increase of 3.7% to 4.8% over the frozen base network. The additional FLOPs are negligible, below 0.2% and within measurement noise, because the parallel branch operates at one eighth of the input resolution on a small number of boundary channels. Peak inference memory stays below 600 MB even at full resolution for the largest variant. This follows directly from our design. Because the base backbone is frozen, no gradients, optimizer states, or replay buffers are kept for it during incremental learning, and the only parameters that grow with each new task belong to the lightweight parallel modules.

Real-time inference. Adding the parallel branch reduces inference speed by only 7.8% to 8.6% across the three variants. All configurations stay firmly in the real-time regime. PILOT runs at 91.5, 71.9, and 57.1 FPS for the S, M, and L backbones, which corresponds to per-frame latencies of 10.93, 13.91, and 17.52 ms. This ranges from roughly three times the conventional 30 FPS real-time threshold for the smallest variant to nearly twice the threshold for the largest. The modest speed reduction is not caused by extra arithmetic, since the FLOPs increase is negligible. On a parallel accelerator, the cost is dominated by the kernel launch overhead, the extra feature upsampling, and the separate prediction head

of the parallel branch. The added modules therefore have low arithmetic intensity, and the framework keeps the high frame rates required for autonomous perception.

Comparison with heavier backbones. The continual learning methods we compare against in Section 4.2 all run on DeepLabV3 with a ResNet-101 backbone. Measured under identical conditions, this backbone reaches only 13.4 FPS at full resolution, which is well below the 30 FPS real-time threshold, with a latency of 74.7 ms, about 1933 GFLOPs, and 1146 MB of memory. PILOT with PIDNet-M is therefore about five times faster, uses roughly eleven times fewer FLOPs, and needs less than half the memory and parameters while still supporting continual adaptation. This gap is the practical reason a real-time backbone is required for incremental learning after deployment, and it is hardware independent because both models are timed on the same A100. We also note that PIDNet sits on the accuracy and speed frontier of recent real-time segmentation models such as DDRNet [16] and STDC [15], neither of which provides a mechanism for continual adaptation, which is the capability our framework adds.

Training cost. Because backpropagation is restricted to the lightweight parallel modules, each incremental learning step is inexpensive. A single training iteration at a 512×1024 crop with a batch size of two, covering the forward pass, loss, and backpropagation through the unfrozen modules, takes 15.4, 16.0, and 17.8 ms for the S, M, and L variants. The peak training memory is 216, 382, and 454 MB, which stays low because the gradients and optimizer state are kept only for the small parallel branch. This confirms that PILOT enables incremental adaptation after deployment without the heavy retraining cost associated with full-network fine tuning or multi-scale distillation pipelines.

4.7. Qualitative Analysis

To provide visual insight into the model's incremental learning capability, we present the qualitative segmentation results in Figure 7. We visualize the predictions across three model capacities (PIDNet-S, M, and L) as the model sequentially learns four new classes: Bus (Step 16), Train (Step 17), Motorcycle (Step 18), and Bicycle (Step 19).

As observed in the figure, our method demonstrates high plasticity throughout the incremental process. At Step 2, the model successfully identifies the “Bus” (highlighted in dark blue), effectively distinguishing it from the surrounding traffic. As the learning progresses to Step 5, the model accurately segments the “Bicycle” (dark red). This is a particularly challenging scenario, as bicycles possess thin structural elements that often visually overlap with the “Person” class. Despite this complexity, our method generates sharp boundaries for the new objects without eroding the segmentation quality of the previously learned categories.

Equally important is the stability of the base classes. In standard fine-tuning approaches, the introduction of new categories often leads to catastrophic forgetting, where the model begins to confuse previously learned textures with the new object, as shown in Figure 1. However, looking at the progression from Step 1 to Step 5, the segmentation masks for base classes such as “Vegetation,” “Car,” and “Person” remain consistent and distinct. Comparing the model variants, the PIDNet-M backbone (middle row) exhibits the most stable boundaries. While PIDNet-S misses some fine details on the bicycle and PIDNet-L introduces slight noise in the background, PIDNet-M maintains a clean separation between the new bicycle class and the existing person class, visually confirming our quantitative finding that the medium-capacity backbone offers the optimal balance for this task.

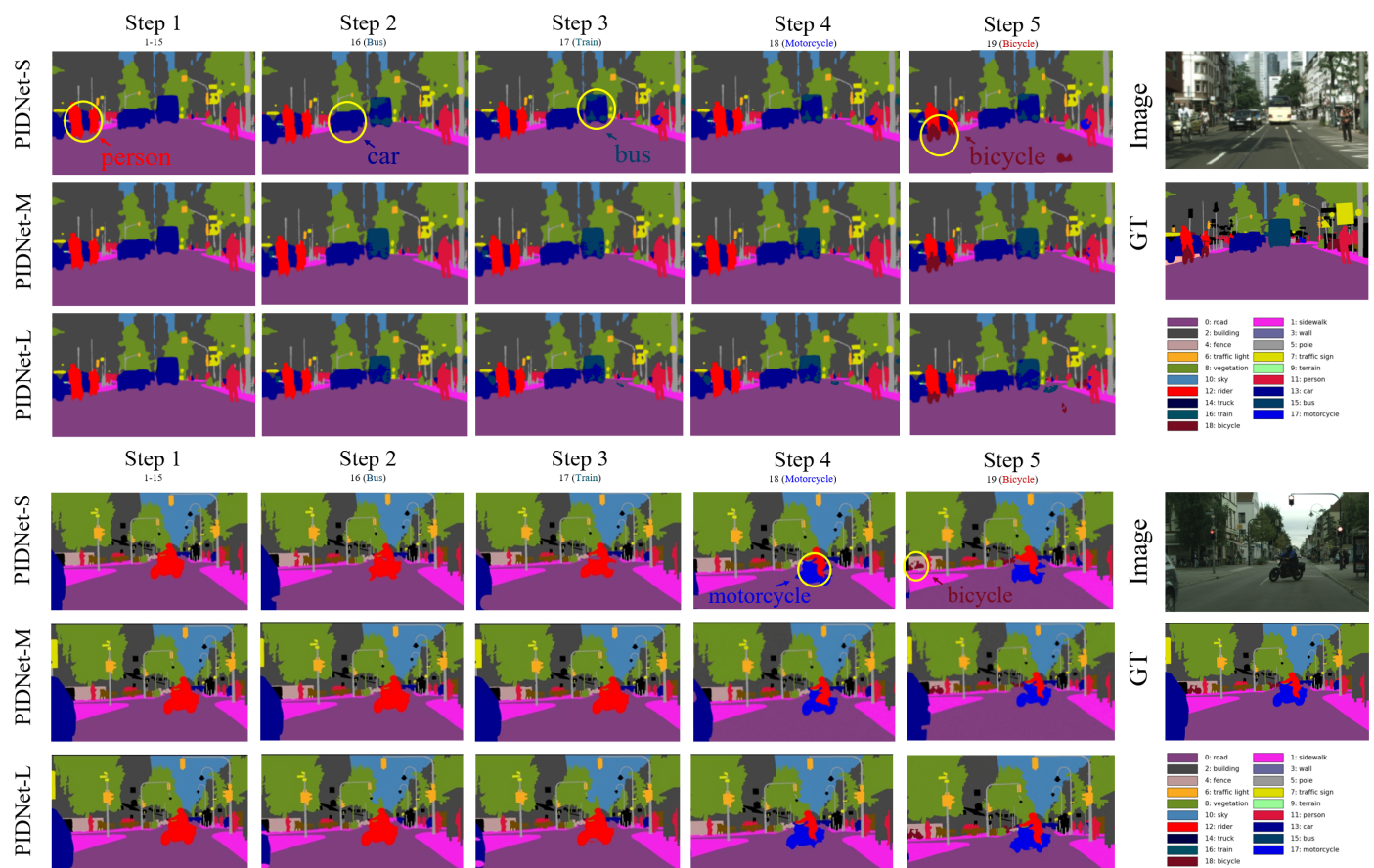


Figure 7. Qualitative evolution of incremental segmentation. The rows correspond to different model capacities (PIDNet-S, M, and L). The columns display the model's predictions at each incremental step: Step 1 (Base 15 classes), Step 2 (+Bus), Step 3 (+Train), Step 4 (+Motorcycle), and Step 5 (+Bicycle). The final column shows the original Input Image and the Ground Truth (GT). Note how the models successfully segment the new “Bicycle” class (Step 5) while preserving the boundaries of the “Person” and “Car” classes learned in Step 1.

5. Discussion

The central finding of this paper is that the balance between stability and plasticity in real-time semantic segmentation can be achieved structurally rather than through data replay or knowledge distillation. By freezing the backbone and confining all plasticity to a lightweight parallel branch on the boundary pathway, PILOT learns new categories without disturbing the contextual representations that encode previously learned classes, which is the mechanism underlying its strong resistance to catastrophic forgetting.

Furthermore, the capacity analysis reveals an important dynamic regarding model scaling in continual learning. Although larger models are generally assumed to perform better, the medium-capacity backbone (PIDNet-M) consistently outperforms the larger PIDNet-L. As quantified in Section 4.3 and Table 5, this is because the larger trainable branch of PIDNet-L overfits the limited novel-class data, whereas PIDNet-M provides the structural balance needed to learn new boundary features while resisting catastrophic interference.

Taken together, these results show that the framework retains a mean IoU of 77.38% on the base classes while reaching 71.87% overall after four incremental steps, and it achieves this without the multi-scale distillation or memory-intensive replay that conventional methods built on heavy backbones such as ResNet-101 require. The added plasticity

therefore comes at minimal computational cost, which is what makes continual adaptation practical under the latency budgets of real-time deployment.

Limitations and failure cases. Two limitations define the scope of this paper. First, our protocol introduces a single novel class per incremental step. The New Class Head is a multi-channel module that can be widened to emit several novel classes at once (Section 3), so $|\mathcal{C}_t| = 1$ reflects our evaluation rather than a constraint of the method; the simultaneous-addition setting, however, is not tested here. Adding several classes together would stress two factors that the one-class protocol does not expose: the single global routing threshold, which already varies across classes (Table 4), becomes a coarser compromise; and a batch of classes with very different frequency and scale must share one lightweight branch, which can bias it toward the more common class.

Second, because the New Class Head is trained only on pixels of the current novel class, a category that closely resembles a base class, for example a bus relative to a truck or a motorcycle relative to a bicycle, can trigger false positives on the look-alike region. This effect is small in our four-class sequence, where only 0.09% of base pixels are mis-routed and 85.6% of routed pixels are correct, but it is expected to grow as added classes become more entangled with the frozen base classes. More fundamentally, the framework inherits the fixed capacity of the frozen backbone: classes that are poorly separable in its feature space, or extremely rare ones such as Train, are bounded in achievable accuracy regardless of the parallel branch. These settings are left to future work.

Future work. Despite the strong results, there are still several directions worth exploring in future work. First, the proposed framework is currently built on top of a convolutional dual resolution backbone, where the clear separation between the Proportional, Integral, and Derivative branches makes it natural to attach a Parallel Boundary Branch. A logical next step is to test whether the same idea works beyond this single type of architecture. Transformer-based segmentation models, and other dense prediction tasks such as object detection, also rely on a mix of contextual and high-frequency features. It would be valuable to study how a frozen backbone paired with a small unfrozen branch can be applied to attention-based encoders or to detection heads that need to keep learning new object categories over time. This would help confirm whether the stability and plasticity benefits seen here come from a general design principle rather than from something specific to PIDNet. Second, the current evaluation is mainly focused on driving datasets such as Cityscapes, where the classes and scenes are closely tied to vehicles, roads, and urban environments. Early experiments on Pascal VOC show that the method can transfer in principle, but the accuracy on more diverse classes is not yet as strong as the in-domain results. Future work will therefore aim to extend the method to more general and varied segmentation datasets. This includes looking at how the boundary cues from the Derivative branch behave when new classes are non-rigid, deformable, or visually very different from the base classes, and finding ways to keep the parallel branch lightweight while still handling the wider range of objects needed for use beyond autonomous driving. Third, our framework uses a single global confidence threshold for routing. As shown in Table 4, the New Class Head is so discriminative for every class ($\text{AUC} > 0.99$) that one global threshold already routes all classes reliably, so per-class tuning is not required for the settings studied here. A natural extension is an adaptive or class-conditioned threshold that is calibrated automatically, which could offer a small additional gain on visually similar classes such as Bicycle and Motorcycle and, more importantly, keep routing reliable when the operating point drifts under distribution shift, for example across cities, weather, or sensor changes. Future work will therefore study confidence-calibration techniques that preserve this reliability without re-tuning τ on held-out data from each new domain.

6. Conclusions

In conclusion, PILOT successfully bridges the gap between advanced continual learning strategies and the strict latency constraints of lightweight segmentation models. Unlike traditional frameworks that rely on heavy offline architectures, the proposed method preserves high frame rates and low inference latency. These characteristics are absolutely essential for real-time processing. It offers a highly efficient and deployment ready solution for real-world autonomous driving systems. For everyday applications, this means autonomous vehicles, delivery robots, and smart city cameras can continuously learn to recognize new things on the fly, including novel vehicles and temporary road barriers. By maintaining real-time perception speeds while entirely eliminating the need for costly offline retraining, this framework ensures that autonomous systems can safely and instantly adapt to rapidly changing environments.

Author Contributions: Conceptualization, Y.Z. and P.S.; methodology, Y.Z. and P.S.; software, Y.Z.; validation, Y.Z.; formal analysis, Y.Z. and P.S.; investigation, Y.Z.; resources, P.S. and Y.L.; data curation, Y.Z.; writing—original draft preparation, Y.Z.; writing—review and editing, Y.Z., P.S., T.Y. and Y.L.; visualization, Y.Z.; supervision, P.S., T.Y. and Y.L.; project administration, Y.L.; funding acquisition, Y.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Cyber-CARE Center.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The Cityscapes dataset used in this paper is publicly available at <https://www.cityscapes-dataset.com/> (accessed on 22 June 2026). The source code implementing our PILOT framework is available at <https://github.com/U1overground/PILOT> (accessed on 22 June 2026).

Conflicts of Interest: The authors declare no conflicts of interest.

References

- Long, J.; Shelhamer, E.; Darrell, T. Fully Convolutional Networks for Semantic Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 3431–3440. [\[CrossRef\]](#)
- Cordts, M.; Omran, M.; Ramos, S.; Rehfeld, T.; Enzweiler, M.; Benenson, R.; Franke, U.; Roth, S.; Schiele, B. The Cityscapes Dataset for Semantic Urban Scene Understanding. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 3213–3223. [\[CrossRef\]](#)
- Parisi, G.I.; Kemker, R.; Part, J.L.; Kanan, C.; Wermter, S. Continual Lifelong Learning with Neural Networks: A Review. *Neural Netw.* **2019**, *113*, 54–71. [\[CrossRef\]](#) [\[PubMed\]](#)
- Rebuffi, S.A.; Kolesnikov, A.; Sperl, G.; Lampert, C.H. iCaRL: Incremental Classifier and Representation Learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 2001–2010. [\[CrossRef\]](#)
- McCloskey, M.; Cohen, N.J. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychol. Learn. Motiv.* **1989**, *24*, 109–165. [\[CrossRef\]](#)
- French, R.M. Catastrophic forgetting in connectionist networks. *Trends Cogn. Sci.* **1999**, *3*, 128–135. [\[CrossRef\]](#) [\[PubMed\]](#)
- Kirkpatrick, J.; Pascanu, R.; Rabinowitz, N.; Veness, J.; Desjardins, G.; Rusu, A.A.; Milan, K.; Quan, J.; Ramalho, T.; Grabska-Barwinska, A.; et al. Overcoming catastrophic forgetting in neural networks. *Proc. Natl. Acad. Sci. USA* **2017**, *114*, 3521–3526. [\[CrossRef\]](#) [\[PubMed\]](#)
- Li, Z.; Hoiem, D. Learning without forgetting. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *40*, 2935–2947. [\[CrossRef\]](#) [\[PubMed\]](#)
- Cermelli, F.; Mancini, M.; Rota Bulò, S.; Ricci, E.; Caputo, B. Modeling the Background for Incremental Learning in Semantic Segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 9233–9242. [\[CrossRef\]](#)
- Tan, G.; Jin, Y. A Semantic Segmentation Method for Road Sensing Images Based on an Improved PIDNet Model. *Electronics* **2025**, *14*, 871. [\[CrossRef\]](#)

11. Paszke, A.; Chaurasia, A.; Kim, S.; Culurciello, E. ENet: A Deep Neural Network Architecture for Real-Time Semantic Segmentation. *arXiv* **2016**, arXiv:1606.02147.
12. Zhao, H.; Qi, X.; Shen, X.; Shi, J.; Jia, J. ICNet for Real-Time Semantic Segmentation on High-Resolution Images. In *Proceedings of the European Conference on Computer Vision (ECCV)*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 405–420. [[CrossRef](#)]
13. Yu, C.; Wang, J.; Peng, C.; Gao, C.; Yu, G.; Sang, N. BiSeNet: Bilateral Segmentation Network for Real-Time Semantic Segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 325–341. [[CrossRef](#)]
14. Poudel, R.P.; Liwicki, S.; Cipolla, R. Fast-SCNN: Fast Semantic Segmentation Network. In *Proceedings of the British Machine Vision Conference (BMVC)*, Cardiff, UK, 9–12 September 2019.
15. Fan, M.; Lai, S.; Huang, J.; Wei, X.; Chai, Z.; Luo, J.; Wei, X. Rethinking BiSeNet For Real-time Semantic Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Nashville, TN, USA, 20–25 June 2021; pp. 9716–9725. [[CrossRef](#)]
16. Hong, Y.; Pan, H.; Sun, W.; Jia, Y. Deep dual-resolution networks for real-time and accurate semantic segmentation of road scenes. *arXiv* **2021**, arXiv:2101.06085.
17. Xu, J.; Zhang, X.; Jin, X.; Chen, Y.; Ye, X. PIDNet: A Real-time Semantic Segmentation Network Inspired from PID Control. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Vancouver, BC, Canada, 17–24 June 2023; pp. 19529–19539. [[CrossRef](#)]
18. Takikawa, T.; Acuna, D.; Jampani, V.; Fidler, S. Gated-scnn: Gated shape cnns for semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 5229–5238. [[CrossRef](#)]
19. Kirillov, A.; Wu, Y.; He, K.; Girshick, R. PointRender: Image Segmentation as Rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA, 13–19 June 2020; pp. 9799–9808. [[CrossRef](#)]
20. Ai, Y.; Sun, Q.; Xi, Z.; Li, N.; Dong, J.; Wang, X. Stereo SLAM in Dynamic Environments Using Semantic Segmentation. *Electronics* **2023**, *12*, 3112. [[CrossRef](#)]
21. Michieli, U.; Zanuttigh, P. Incremental Learning Techniques for Semantic Segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, Seoul, Republic of Korea, 27–28 October 2019. [[CrossRef](#)]
22. Douillard, A.; Chen, Y.; Dapogny, A.; Cord, M. PLOP: Learning Without Forgetting for Continual Semantic Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Nashville, TN, USA, 20–25 June 2021; pp. 4039–4049. [[CrossRef](#)]
23. Park, S.H.; Noh, J.Y.; Ham, B. SSUL: Semantic Segmentation with Unknown Labels for Exemplar-based Class-Incremental Learning. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, Online, 6–14 December 2021; Volume 34, pp. 10919–10930.
24. Zhao, D.; Yuan, B.; Shi, Z. Inherit with Distillation and Evolve with Contrast: Exploring Class-Incremental Semantic Segmentation without Exemplar Memory. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *45*, 11932–11947. [[CrossRef](#)] [[PubMed](#)]
25. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778. [[CrossRef](#)]
26. Chen, L.C.; Zhu, Y.; Papandreou, G.; Schroff, F.; Adam, H. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 801–818. [[CrossRef](#)]
27. Zhang, C.; Wang, L.; Zhang, J. Representation Compensation Networks for Continual Semantic Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, New Orleans, LA, USA, 18–24 June 2022; pp. 7053–7064. [[CrossRef](#)]
28. Zhou, H.; Kim, D.; Peng, X.; Lee, T. Uncertainty-Aware Contrastive Distillation for Incremental Semantic Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *45*, 2567–2581. [[CrossRef](#)] [[PubMed](#)]
29. Goswami, D.; Schuster, R.; van de Weijer, J.; Stricker, D. Attribution-aware Weight Transfer: A Warm-Start Initialization for Class-Incremental Semantic Segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, Waikoloa, HI, USA, 2–7 January 2023; pp. 3195–3204. [[CrossRef](#)]
30. Baek, D.; Oh, Y.; Lee, S.; Lee, J.; Ham, B. Decomposed knowledge distillation for class-incremental semantic segmentation. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 10380–10392. [[CrossRef](#)]
31. Hindel, J.; Cattaneo, D.; Valada, A. Taxonomy-Aware Continual Semantic Segmentation in Hyperbolic Spaces for Open-World Perception. *arXiv* **2024**, arXiv:2407.18145.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.