AIDE: An Automatic Data Engine for Object Detection in Autonomous Driving

Mingfu Liang¹  Jong-Chyi Su²  Samuel Schulter²  Sparsh Garg²  Shiyu Zhao³
Ying Wu¹  Manmohan Chandraker²,⁴
¹ Northwestern University  ² NEC Laboratories America  ³ Rutgers University  ⁴ UC San Diego

Abstract

Autonomous vehicle (AV) systems rely on robust perception models as a cornerstone of safety assurance. However, objects encountered on the road exhibit a long-tailed distribution, with rare or unseen categories posing challenges to a deployed perception model. This necessitates an expensive process of continuously curating and annotating data with significant human effort. We propose to leverage recent advances in vision-language and large language models to design an Automatic Data Engine (AIDE) that automatically identifies issues, efficiently curates data, improves the model through auto-labeling, and verifies the model through generation of diverse scenarios. This process operates iteratively, allowing for continuous self-improvement of the model. We further establish a benchmark for open-world detection on AV datasets to comprehensively evaluate various learning paradigms, demonstrating our method’s superior performance at a reduced cost.

1. Introduction

Autonomous vehicles (AVs) operate in an ever-changing world, encountering diverse objects and scenarios in a long-tailed distribution. This open-world nature poses a significant challenge for AV systems since it is a safety-critical application where reliable and well-trained models must be deployed. The need for continuous model improvement becomes apparent as the environment evolves, demanding adaptability to handle unexpected events. Despite the wealth of data collected on the road every minute, its effective utilization remains low due to challenges in discerning which data to leverage. While solutions exist for this in industry [1, 2], they are often trade secrets and presumably require significant human effort. Hence, developing a comprehensive automated data engine can lower entry barriers for the AV industry.

Designing automated data engines can be challenging, but the existence of Vision-Language Models (VLMs) and Large Language Models (LLMs) allows new avenues to these hard problems. A traditional data engine can be broken down into finding issues, curating and labeling data, model training, and evaluation, all of which can benefit from automation. In this paper, we propose an Automatically Improving Data Engine (called AIDE) that leverages VLMs and LLMs to automate the data engine. Specifically, we use VLMs to identify the issue, query relevant data, auto-label data, and verify together with LLMs. The high-level steps are shown in Fig. 1 top.

In contrast to traditional data engines that rely heavily on extensive human labeling and intervention, AIDE automated the process by utilizing pre-trained VLMs and LLMs. Different from other confidential solutions in industry [1, 2], we provide our efficient solutions to lower the entry barrier. While open-vocabulary object detection (OVOD) methods [3, 4] do not require any human annotations, they are a good starting point for detecting novel objects but their...
performances fall short on AV datasets compared to supervised methods. Another line of research on minimizing labeling costs is semi-supervised learning [5, 6] and active learning [7–10]. Although they generate pseudo-labels, the vast amount of data collected on the road is still not fully utilized, in contrast with our method which leverages pre-trained VLMs and LLMs for better data utilization.

The detailed steps of AIDE are shown in Fig. 2. In the Issue Finder, we use a dense captioning model to describe the image in detail, then match if the objects in the description are included in the label spaces or the predictions. This is based on the reasonable but previously unexploited assumption that large image captioning models are more robust starting points in zero-shot settings than OVOD (Tab. 3). The next step is to find relevant images that could contain the novel category using our Data Feeder. We find that VLM gives more accurate image retrieval than using image similarity to retrieve images (Tab. 4). We then use our existing label space plus the novel category to prompt the OVOD method, i.e., OWL-v2 [11], to generate predictions on the queried images. To filter these pseudo predictions, we use CLIP to perform zero-shot classification on the pseudo-boxes to generate pseudo-labels for the novel categories. Last, we exploit the LLM, e.g., ChatGPT [12], in Verification to generate diverse scene descriptions given the novel objects. Given the generated description, we again use VLM to query relevant images to evaluate the updated model. To ensure the correctness, we ask humans to review if the predictions of the novel categories are correct. If it is not, we ask humans to provide ground-truth labels, which are used to further improve the model. (Fig. 6)

To verify the effectiveness of our AIDE, we propose a new benchmark on existing AV datasets to comprehensively compare our AIDE with other paradigms. With our Issue Finder, Data Feeder, and Model Updater, we bring 2.3% Average Precision (AP) improvement on the novel categories compared with OWL-v2 without any human annotations and also surpass OWL-v2 by 8.9% AP on known categories (Tab. 1). We also show that with a single round of Verification, our automatic data engine can further bring 2.2% AP on novel categories without forgetting the known categories, as shown in Fig. 1. To summarize, our contributions are two-fold:

• We propose a novel design paradigm for an automatic data engine for autonomous driving as automatic data querying and labeling with VLM and continual learning with pseudo labels. When scaling up for novel categories, this approach achieves an excellent trade-off between detection performance and data cost.
• We introduce a new benchmark to evaluate such automated data engines for AV perception that allows combined insights across multiple paradigms of open vocabulary detection, semi-supervised, and continual learning.

2. Related Works

Data Engine for Autonomous Vehicles (AV) Exploiting large-scale data collected by AV is crucial to speed up the iterative development of the AV system [13]. Existing literature mostly focuses on developing general [14, 15] learning engines or specific [16] data engines, and most of them [17, 18] mainly focus on the model training part. However, a fully functional AV data engine requires issue identification, data curation, model retraining, verification, etc. A thorough examination reveals a lack of systematic research papers or literature that delves deeply into AV data engines in academia, where a recent survey [13] also underscores the lack of study in this context. On the other hand, existing solutions [1, 2] for AV data systems mainly rely on the design of data infrastructure and still need lots amount of human effort and intervention, thus limiting their maintenance simplicity, affordability, and scalability. In contrast, the present paper exploits the burgeoning progress of vision language models (VLMs) [19–21] to design our data engine, where their strong open-world perception capability largely improves our engine’s extendability and makes it more affordable to scale up our AVs on detecting novel categories. To our best knowledge, this paper is also the first work that provides a systematic design of data engines for AVs with the integration of VLMs.

Novel Object Detection Conventional 2D object detection has made enormous progress [22, 23] in the last decades, while its closed-set label space makes unseen category detection infeasible. On the other hand, open-vocabulary object detection (OVOD) [4, 24–39] methods promises to detect anything by a simple text prompting. However, their performances are still inferior to closed-set object detection since they must balance the specificity of pre-trained categories and the generalizability of unseen categories. To scale up the capacity of open-vocabulary detector (OVD), recent works either pre-train OVD with weak annotations (e.g., image captions) [40], or perform self-training on daily object datasets [41, 42] or web-scale datasets [4, 43]. However, balancing the trade-off between improving the novel categories while mitigating the catastrophic forgetting of the known categories is still an open problem that has not been resolved [11], making it hard to adapt to task-specific applications like autonomous driving.

On the other hand, limited research has focused on novel object detection for AVs. This is especially crucial because a false-negative detection of unseen objects may result in fatal consequences for AVs. Existing OVOD methods mostly benchmark on datasets of general objects [42, 44] while putting little attention on AV datasets [45–50]. Different from the pursuit of generality in OVOD, perception in AVs has its domain concerns oriented from the image-capturing process by on-car cameras and the object categories due to the scene prior (e.g., road/street objects), which demands
Figure 2. Our design of the automatic data engine includes Issue Finder, Data Feeder, Model Updater, and Verification. The Issue Finder automatically identifies novel categories using the dense captioning model. In the Data Feeder, we employ VLMs to efficiently search for relevant data for training, significantly reducing the inference time for generating pseudo-labels in the subsequent steps and filtering out unrelated images for training. The model is updated in the Model Updater using auto-labeling by VLMs, enabling the recognition of novel categories without incurring any labeling costs. To verify the model, in Verification, we use LLMs to generate descriptions of variations in scenarios and then assess predictions on images queried by VLMs.

3. Method

This section demonstrates our proposed AIDE, composed of four components: Issue Finder, Data Feeder, Model Updater, and Verification. The Issue Finder automatically identifies missing categories in the existing label space by comparing detection results and dense captions given an image. This triggers the Data Feeder to perform text-guided retrieval for relevant images from the large-scale image pool collected by AVs. The Model Updater then automatically labels queried images and continuously trains the novel category with pseudo-labels on the existing detector. The updated detector is then passed to the Verification module to evaluate under different scenarios and trigger a new iteration if needed. We outline our systematic design in Fig. 2.

3.1. Issue Finder

Given the large amount of unlabeled data collected by AVs in daily operation, identifying the missing category of existing label space is difficult as it requires humans to extensively compare detection results and dense captions given an image. This triggers the Data Feeder to perform text-guided retrieval for relevant images from the large-scale image pool collected by AVs. The Model Updater then automatically labels queried images and continuously trains the novel category with pseudo-labels on the existing detector. The updated detector is then passed to the Verification module to evaluate under different scenarios and trigger a new iteration if needed. We outline our systematic design in Fig. 2.
Figure 3. Examples of the Issue Finder. We use Otter [20] to generate detailed descriptions of an image, then identify the novel category that is missing in the label space (shown in red).

Figure 4. Visualization of the queried images from Data Feeder on three novel categories.

Figure 5. Our two-stage pseudo-labeling for Model Updater: generate boxes by zero-shot detection and label by CLIP filtering.

Table 4. Instead, our Data Feeder utilizes the VLMs to perform text-guided image retrieval on the image pool to query for relevant images related to the novel categories. We consider BLIP-2 [21] given its strong open-vocabulary text-guided retrieval capability. Precisely, given an image and a specific text input, we measure the cosine similarity between their embeddings from BLIP-2 and only retrieve the top-k images for further labeling in our Model Updater. For the text prompt, we experiment with common prompt engineering practice [71] and find that a template like “An image containing {}” can readily provide good precision and recall for the novel categories in practice. Fig. 4 shows some examples of retrieved images.

3.3. Model Updater

The goal of our Model Updater is to make our detector learn to detect novel objects without human annotations. To this end, we perform pseudo-labeling on the images queried by the Data Feeder and then use them to train our detector.

3.3.1 Two-Stage Pseudo-Labeling

Motivated by the previous success in pseudo-labeling for object detection [41], we designed our pseudo-labeling procedure with two parts: box and label generation. Such a two-stage framework can help us better dissect the issue of pseudo-label generation and improve the label generation quality. Box generation aims to identify as many object proposals in the image as possible, i.e., high recall for localizing novel categories, to guarantee a sufficient number of candidates for label generation. To this end, region proposal networks (RPN) pretrained with closed-set label space [41] and the open vocabulary detectors (OVD) [11] can be considered, where the former can localize generic objects while the latter can perform text-guided localization. We observe that the SOTA OVD, i.e., OWL-v2 [11] that has been self-trained on web-scale datasets [43], exhibits a higher recall to localize novel categories compared to the RPN. We conjecture that proposals of RPN may be readily biased toward the pre-trained categories.

Thus, we choose OWL-v2 as our zero-shot detector to get the box proposal. Specifically, we append the novel category name provided by Issue Finder to our existing label space and create the text prompts, then we prompt the
OWL-v2 to inference on an image. Note that we only retain the box proposals and remove the labels from the OWL-v2’s predictions. This is because we empirically find that OWL-v2 can not achieve reliable precision on the novel categories presented in AV datasets, e.g., less than 10% AP averaging over the novel categories in AV datasets [45, 50], while it can get >40% AP on novel categories of LVIS [42] datasets. We conjecture that this performance degradation may come from the domain shift of the images collected in the AV scenario. For instance, the pretraining data of OWL-v2 mainly comes from the daily image captured by humans from a close distance. However, the street objects are always small in the image due to their long distance from the on-car camera, and the aspect ratio of the image presented in AV datasets is relatively large, making OWL-v2 hard to classify the correct label of the object proposals.

Motivated by this insight, we consider conducting another round of label filtering with CLIP [71] to purify the predictions of the OWL-v2 and generate the pseudo labels. Specifically, we pass the box prediction by OWL-v2 to the original CLIP model [71] for zero-shot classification (ZSC), as shown in Fig. 5. To mitigate the potential issue of the aspect ratio mentioned above, we increase the box size to crop the image and then send the cropped image patch to CLIP for ZSC. This can involve more scene contextual information to help the CLIP better differentiate between the novel and known categories. Regarding the label space for CLIP to do zero-shot classification, we first create a base label space, which is a combination of the label space from datasets we have pre-trained and COCO [44], to ensure that we can mostly cover daily objects that would probably be present in the street. The base label space will automatically extend when the Issue Finder identifies novel categories not in the base label space.

3.3.2 Continual Training with Pseudo-labels

Directly training our existing detector on the pseudo-labels of novel categories presents a challenge, as these labels may lead the detector to overfit and catastrophically forget the known categories. The issue arises because the unlabeled data can contain both novel and known categories that the detector has previously learned. Without labels for those known categories and only having labels for novel categories, the model may incorrectly suppress predictions for known categories, focusing solely on predicting novel categories. As training progresses, the known categories gradually fade from memory. To address this issue, we draw inspiration from existing self-training strategies and include the pseudo-labels of the known categories that have been trained on. Consequently, our existing detector is updated with the pseudo-labels of both novel and known categories. To obtain pseudo-labels for the known categories, we first use our detector to infer data before applying OWL-v2 to the data. Empirically, we find that including pseudo-labels for known categories helps the model distinguish between known and novel categories, boosting the performance of novel categories and mitigating the catastrophic forgetting issues associated with known categories. Additionally, acknowledging that pseudo-labels for both known and novel categories may not be perfect, we filter the pseudo-labels. For known categories, we only use pseudo-labels with high predicted confidence from our detector. For novel categories, we have already incorporated CLIP to filter pseudo-labels, as mentioned in Section 3.3.1.

3.4. Verification

The Verification step aims to evaluate whether the updated detector can detect the novel categories under different scenarios, to ensure the model can handle unexpected or unseen scenarios. To this end, we prompt the ChatGPT [12] with the name of novel categories to generate diverse scene descriptions. These descriptions contain variations of the scenarios, such as different appearances of the objects, surrounding objects, time of the day, weather conditions, etc. For each scene description, we again use BLIP-2 to query relevant images, which are used to test the model’s robustness. To ensure the correctness, we ask humans to review if the predictions for the novel categories are correct. If the predictions are correct, the detector has passed the unit test. Otherwise, we ask humans to provide the ground-truth label, which can be used to further improve the model. Compared to existing solutions that have humans manually examine the model prediction one by one, our Verification ex-
Table 1. Cost and accuracy for fully-supervised, open-vocabulary object detection, semi-supervised learning, and our data engine (AIDE) to detect one novel category from Mapillary and nuImages. We initialize Semi-SL and ours with the same detector.

Table 2. Per-category accuracy (AP %) on novel categories with different methods.

4. Experiments

4.1. Experimental Setting

Datasets and Novel Categories Selection In reality, the AV system can hardly train with a single source of data, e.g., AVs may operate in various locations in the world to collect data. To simulate such a nature faithfully, we leverage the existing AV datasets to jointly train our closed-set detector, including Mapillary [50], Cityscapes [47], nuImages [45], BDD100k [49], Waymo [46], and KITTI [48]. We use this pretrained detector as the initialization for the supervised training, Semi-SL, and our AIDE for a fair comparison. There are 46 categories in total after combining the label spaces. To simulate the novel categories and ensure that the selected categories are meaningful and crucial for AV in the street, we choose 5 categories as novel categories: “motorcyclist” and “bicyclist” from Mapillary, “construction vehicle”, “trailer”, and “traffic cone” from nuImages. The rest 41 categories are set as known. We remove all the annotations for these categories in our joint datasets and also remove the related categories with similar semantic meanings, e.g., “bicyclist” vs “cyclist”. We attach more details of the dataset statistics in the supplementary material.

Methods for Comparison To our knowledge, there is little work about the systematic design for automatic data engines tailored to the novel object detection for AV systems. Thus, it is hard to identify a comparable counterpart for our AIDE. To this end, we dissect our evaluation into two parts: (1) compare to alternative detection methods and learning paradigms on the performance of novel object detection; (2) ablation study and analysis of each step of the automatic data engine. For (1), as our AIDE can enable the detector to detect novel categories without any labels, we first compare our method with the zero-shot OVOD methods on novel categories’ performance. Moreover, to show the efficiency and effectiveness of our AIDE in reducing label cost, we further compare with semi-supervised learning (Semi-SL) and fully supervised learning that trains the detector with different ratios of ground-truth labels. Specifically, we compare our data engine to state-of-the-art (SOTA) OVOD methods like OWL-v2 [11], OWL-ViT [4], and Semi-SL methods like Unbiased Teacher [5, 6].

Experimental Protocols We treat each of the five selected classes as novel classes and conduct experiments separately to simulate the scenario that one novel class has been identified at a time by our Issue Finder. For Semi-SL methods, we provide different numbers of ground-truth images for training. Each image could contain one or multiple objects of the novel category. We evaluate all comparison methods on the dataset of the novel category for a fair comparison.

Evaluation As our AIDE automates the whole data curation, model training, and verification process for the AV system, we are interested in how our engine can strike a balance between the cost of searching and labeling images
and the performance on novel object detection. We measure the human labeling costs [72] and also the GPU inference costs [73], i.e., the usage of VLMs/LLMs in our AIDE and training the model with pseudo labeled for our AIDE or with ground-truth labels for comparison methods, denoted as ‘Labeling + Training Cost’ in Fig. 1. The labeling cost for a bounding box is $0.06 [72], and the GPU cost is $1.1 per hour [73]. The cost of ChatGPT is negligible (< $0.01).

**Experimental Details**

Given the real-time requirement for inference, we choose the Fast-RCNN [22] as our detector instead of OVOD methods like OWL-ViT [4] as the FPS for OWL-ViT is only 3. We run our AIDE to iteratively scale up its capability of detecting novel objects. For multi-dataset training, we follow the same recipe from [74]. For each novel category, we train for 3000 iterations with the learning rate of 5e-4, and we use the same hyperparameter for all the comparison methods if they require training. We attach our full experimental details in the supplementary material.

### 4.2. Overall Performance

In this section, we provide the overall performance of novel object detection after running our AIDE for a complete cycle. Our results are shown in Fig. 1 and Tab. 1. Compared to the SOTA OVOD method, OwL-v2 [11], our method outperforms by 2.3%AP on novel categories and 8.7%AP on known categories, showing that our AIDE can benefit from mining the open-vocabulary knowledge from OVOD method. This is due to our simple yet effective continual training strategy described in Section 3.3.2. Moreover, our AIDE suffers much less from catastrophic forgetting compared to Semi-SL methods, since current Semi-SL methods for object detection do not contain continual learning settings. Existing works on continual semi-supervised learning [67, 70] only consider image classification and are not applicable to object detection. Combining our AIDE with and without the Data Feeder makes it apparent that our Data Feeder can sufficiently reduce the inference time cost as the Data Feeder can pre-filter irrelevant images, and the Model Updater only needs to assign pseudo-labels on a small number of relevant images. Tab. 1 shows that pre-filtering leads to better AP on novel categories.

### 4.3. Analysis on AIDE

In the following subsections, we will dissect each part of our AIDE to validate our design choice.

#### 4.3.1 Issue Finder

As mentioned in Section 3.1, the main goal of our Issue Finder is to automatically identify categories that do not exist in our label space. To this end, we evaluate the success rate of automatically identifying the novel categories. We find that dense captioning models can automatically predict

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Category Name</th>
<th>Dense Captioning Precision (%)</th>
<th>OVOD AP50 (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mapillary</td>
<td>motorcyclist</td>
<td>83.3</td>
<td>9.5</td>
</tr>
<tr>
<td>Mapillary</td>
<td>bicyclist</td>
<td>89.5</td>
<td>1.6</td>
</tr>
<tr>
<td>nuImages</td>
<td>const. vehicle</td>
<td>65.6</td>
<td>12.9</td>
</tr>
<tr>
<td>nuImages</td>
<td>trailer</td>
<td>24.7</td>
<td>7.1</td>
</tr>
<tr>
<td>nuImages</td>
<td>traffic cone</td>
<td>87.9</td>
<td>60.3</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td></td>
<td>70.2</td>
<td>18.3</td>
</tr>
</tbody>
</table>

Table 3. Comparing with using OVOD to identify and localize novel categories, Dense Captioning better predicts missing categories more reliably in our Issue Finder.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Category</th>
<th>Image similarity</th>
<th>VLM Retrieval</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mapillary</td>
<td>motorcyclist</td>
<td>22.6</td>
<td>19.0</td>
</tr>
<tr>
<td>Mapillary</td>
<td>bicyclist</td>
<td>17.9</td>
<td>28.8</td>
</tr>
<tr>
<td>nuImages</td>
<td>const. vehicle</td>
<td>14.2</td>
<td>51.2</td>
</tr>
<tr>
<td>nuImages</td>
<td>trailer</td>
<td>10.5</td>
<td>23.3</td>
</tr>
<tr>
<td>nuImages</td>
<td>traffic cone</td>
<td>29.5</td>
<td>47.3</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td></td>
<td>18.9</td>
<td>33.9</td>
</tr>
</tbody>
</table>

Table 4. Ablation studies of the Data Feeder. We report accuracy (%) of the top-k images queried by image similarity search and text-based retrieval with VLM, i.e., CLIP and BLIP-2.

if the image contains the novel categories more precisely, compared to using OVOD methods to identify and localize novel objects when they are given the names of the novel categories, as shown in Tab. 3. Note that the goal here is to only identify the missing categories, hence we choose to use dense captions here and leverage OVOD to help localize the novel object in the later steps.

#### 4.3.2 Data Feeder

The goal of the Data Feeder is to curate relevant data from a large pool of images with high precision. We compare several choices, including image similarity search by CLIP feature, and text-guided image retrieval by VLMs, i.e., BLIP-2 and the CLIP. We report the accuracy of top-k queried images over different categories in Tab. 4, showing that image similarity search is inferior to VLMs. This is because the novel categories can have large intra-class variations, and thus only one image may not be representative of finding sufficient amounts of relevant images. Compared with CLIP, our choice of BLIP-2 performs better on average.

#### 4.3.3 Model Updater

We ablate the design choices for our box and pseudo-label generation. For box generation, we compare our choice of using box proposals from OWL-v2 with using proposals from VL-PLM [41], which generates box proposals by the region proposal network (RPN) of MaskRCNN [75] pre-
Table 5. Ablation of Model Updater on box generation with SAM and VL-PLM, label generation without CLIP filtering, and continual training excluded pseudo labels of known categories.

<table>
<thead>
<tr>
<th>Category</th>
<th>SAM</th>
<th>VL-PLM</th>
<th>w/o CLIP</th>
<th>ex. known</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>motorcyclist</td>
<td>0.5</td>
<td>10.1</td>
<td>3.3</td>
<td>2.8</td>
<td>8.4</td>
</tr>
<tr>
<td>bicyclist</td>
<td>2.8</td>
<td>6.5</td>
<td>3.2</td>
<td>2.1</td>
<td>11.9</td>
</tr>
<tr>
<td>const. vehicle</td>
<td>1.4</td>
<td>4.3</td>
<td>4.0</td>
<td>3.5</td>
<td>5.7</td>
</tr>
<tr>
<td>trailer</td>
<td>0.4</td>
<td>0.4</td>
<td>2.0</td>
<td>1.1</td>
<td>3.7</td>
</tr>
<tr>
<td>traffic cone</td>
<td>14.5</td>
<td>10.4</td>
<td>30.0</td>
<td>30.9</td>
<td>30.7</td>
</tr>
<tr>
<td>Average AP (%)</td>
<td>3.9</td>
<td>6.3</td>
<td>8.5</td>
<td>8.1</td>
<td>12.0</td>
</tr>
</tbody>
</table>

Table 6. Our Verification step can indeed find diverse scenarios. The diversity is measured by the number of distinct images among 100 queried images using descriptions generated by ChatGPT.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Category</th>
<th>Diversity (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mapillary</td>
<td>motorcyclist</td>
<td>57.6</td>
</tr>
<tr>
<td>Mapillary</td>
<td>bicyclist</td>
<td>62.2</td>
</tr>
<tr>
<td>nuImages</td>
<td>const. vehicle</td>
<td>77.0</td>
</tr>
<tr>
<td>nuImages</td>
<td>trailer</td>
<td>82.0</td>
</tr>
<tr>
<td>nuImages</td>
<td>traffic cone</td>
<td>70.4</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td>69.8</td>
</tr>
</tbody>
</table>

4.3.4 Verification

The goal of the Verification is to evaluate the detector’s robustness and to verify the performance under diverse scenarios. Humans only need to examine if the predictions are correct in each scenario which reduces the monitoring cost since the scenarios are diverse and it takes less time to check the predictions than to annotate. To test if the generated scenarios are diverse, we measure the number of unique images among 100 images queried by generated descriptions and repeat the process ten times. As shown in Tab. 6, our Verification can indeed find diverse scenarios, as 69.8% images are distinct on average, even on such small training datasets.

If the prediction is incorrect, we can ask annotators to label the images, which are used to further improve the detector. To this end, we randomly select 10 LLM-generated descriptions, for which top-1 retrieved image (based on BLIP-2 cosine similarity) was predicted incorrectly, and labeled these 10 images to update our detector by Model Updater. As shown in Fig. 7, after updating the model with a few human supervisions, our model can successfully predict the object, e.g., the motorcyclist in the figure, which was mis-detected before. For the overall performance, we achieve 14.2% AP on novel categories, which improves our zero-shot performance by 2.2% AP, while the total cost only increases to $1.59. This is still less than $2.1 of semi-supervised learning, and our AP for known categories remains 26.6% after Verification.

5. Conclusion

We proposed an Automatic Data Engine (AIDE) that can automatically identify the issues, efficiently curate data, improve the model using auto-labeling, and verify the model through generated diverse scenarios. By leveraging VLMs and LLMs, our pipeline reduces labeling and training costs while achieving better accuracies on novel object detection. The process operates iteratively which allows continuous improvement of the model, which is critical for autonomous driving systems to handle expected events. We also establish a benchmark for open-world detection on AV datasets, demonstrating our method’s better performance at a reduced cost. One of the limitations of AIDE is that VLM and LLM can hallucinate in issue finder and verification. Despite the effectiveness of AIDE, for a safety-critical system, some human oversight is always recommended.

Acknowledgements

This work was supported in part by National Science Foundation grant IIS-2007613.
References

[1] Tesla autonomy day, how published = https://www.youtube.com/watch?v=si=bwlmhvsuzhivax. 1, 2

[2] Cruise’s continuous learning machine predicts the unpredictable on san francisco roads, howpublished = https://medium.com/cruise/cruise-continuous-learning-machine-30d60f4c691b. 1, 2


[12] Introducing chatgpt, how published = https://openai.com/blog/chatgpt. 2, 5, 1, 4


[37] Kuniaki Saito, Ping Hu, Trevor Darrell, and Kate Saenko. Learning to detect every thing in an open world. In European


[59] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervi-


[73] GPU price from lambda, howpublished = https://lambdalabs.com/service/gpu-cloud.


AIDE: An Automatic Data Engine for Object Detection in Autonomous Driving

Supplementary Material

A. Verification can Boost AIDE’s Performance

In Verification, humans are asked to verify the predictions on the diverse scenarios generated by LLMs (ChatGPT [12]). If the prediction is incorrect, annotators can give correct bounding boxes, which can be used by AIDE to self-improve the model. In this section, we examine whether these annotations can boost the performance of AIDE. To this end, we train the model after we have collected annotations for 10, 20, and 30 images. However, since we only have a few human annotations collected, directly combining them with a large number of pseudo-labels from the Model Updater will cause issues if we have a uniform sampling rate on the data loader during training.

On the other hand, semi-supervised learning methods like Unbiased Teacher-v1 [5] have demonstrated notable performance on novel categories with minimal annotations, owing to their strong augmentation strategy.

Motivated by this insight, we first use the few labeled images to train an auxiliary model by the strong augmentation strategy as [5] but with 1000 iterations to reduce training costs. This auxiliary model is then used to generate pseudo-labels for the novel categories based on the images initially queried by our Data Feeder, and these are combined with the earlier pseudo-labels generated by our Model Updater for both novel and known categories to fine-tune our detector again in our Model Updater. By doing so, we can obtain more pseudo-labels for novel categories with high quality and alleviate the sampling issue in the data loader. As shown in Fig. 8, our AIDE can be largely improved.

B. More Comparisons between AIDE and OVOD (OWL-v2)

In this section, we demonstrate that AIDE is a general automatic data engine that can enhance different object detectors for novel object detection. Specifically, we replace the closed-set detector (Faster RCNN [77]) with the state-of-the-art (SOTA) open-vocabulary object detection (OVOD) method, OWL-v2.

As shown in Tab. 7, by applying our AIDE on OWL-v2, we can achieve 13.2% AP on average without human annotations, marking a 3.5% improvement over the original OWL-v2 model. However, our default detector is Faster RCNN since it has a faster inference speed, which is favorable for autonomous driving.

In addition, the original OWL-v2 paper [11] proposes a self-training strategy to enhance the OWL-v2 on novel object detection, i.e., directly using the predictions of OWL-v2 with a certain confidence threshold to self-train the OWL-v2. We compare this self-training schedule with our AIDE.

As shown in Tab. 7, the self-training can improve the OWL-v2, but it is still inferior to AIDE 3.1%. This improvement is attributable to our Data Feeder and the CLIP filtering in our Model Updater, which help to minimize irrelevant images for pseudo-labeling and filter out inaccurate OWL-v2 predictions, thereby enhancing the quality of pseudo-labels and the subsequent performance after fine-tuning OWL-v2 with these labels. We will dissect the impact of our Data Feeder and Model Updater on improving the quality of pseudo-label in Sec. D.2 and Tab. 10.

<table>
<thead>
<tr>
<th>Category</th>
<th>OVOD</th>
<th>OWL-v2 ST</th>
<th>AIDE (Ours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>motorcyclist</td>
<td>4.0</td>
<td>5.3</td>
<td>8.4</td>
</tr>
<tr>
<td>bicyclist</td>
<td>0.9</td>
<td>0.8</td>
<td>11.9</td>
</tr>
<tr>
<td>const. vehicle</td>
<td>4.7</td>
<td>5.4</td>
<td>5.7</td>
</tr>
<tr>
<td>trailer</td>
<td>3.6</td>
<td>3.5</td>
<td>3.6</td>
</tr>
<tr>
<td>traffic cone</td>
<td>35.3</td>
<td>35.5</td>
<td>30.7</td>
</tr>
<tr>
<td>Average AP%</td>
<td>9.7</td>
<td>10.1</td>
<td>12.0</td>
</tr>
</tbody>
</table>

Figure 8. We demonstrate that the annotations in the Verification step can boost the performance of AIDE. The numbers next to the data points denote the number of labeled images used by each method. Note that AIDE only introduces labeled images in Verification if an annotator wants to provide the labels when the detector gives incorrect predictions on the test scenarios.

Table 7. Comparison between OWL-v2, OWL-v2 with self-training, and AIDE on improving an existing detector on novel object detection with any human annotations. ST: Self-training using the same strategy in [11].
Table 8. Extending the image pool with the Waymo and BDD100k dataset in Data Feeder can boost the performance of AIDE.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Category</th>
<th>Mapillary / nuImages</th>
<th>+Waymo (39k)</th>
<th>+Waymo (78k)</th>
<th>+Waymo (78k)</th>
<th>+BDD100k (69k)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mapillary</td>
<td>motorcyclist</td>
<td>8.4</td>
<td>9.4</td>
<td>11.1</td>
<td>13.4</td>
<td></td>
</tr>
<tr>
<td>Mapillary</td>
<td>bicyclist</td>
<td>11.9</td>
<td>13.0</td>
<td>15.0</td>
<td>18.4</td>
<td></td>
</tr>
<tr>
<td>nuImages</td>
<td>const. vehicle</td>
<td>5.7</td>
<td>7.3</td>
<td>14.6</td>
<td>19.7</td>
<td></td>
</tr>
<tr>
<td>nuImages</td>
<td>trailer</td>
<td>3.7</td>
<td>3.6</td>
<td>5.1</td>
<td>11.2</td>
<td></td>
</tr>
<tr>
<td>nuImages</td>
<td>traffic cone</td>
<td>30.7</td>
<td>31.6</td>
<td>35.1</td>
<td>36.1</td>
<td></td>
</tr>
<tr>
<td>Average AP(%)</td>
<td></td>
<td>12.0</td>
<td>13.9</td>
<td>16.2</td>
<td>19.8</td>
<td></td>
</tr>
</tbody>
</table>

C. Extending the Image Pool further boosts AIDE’s Performance

Our Data Feeder queries images from either Mapillary [50] or nuImages [45] by default. To verify the scalability of AIDE, we add the Waymo dataset in the database for Data Feeder, i.e., the image pool for querying becomes \{nuImages, Waymo\} or \{Mapillary, Waymo\} for each novel category. Note that the Waymo dataset only contains three coarse labels, i.e., “vehicle”, “pedestrian”, and “cyclist”, as shown in Tab. 11. Therefore it is uncertain whether novel categories such as “motorcyclist”, “construction vehicle”, “trailer”, and “traffic cone” are present in the Waymo dataset. For “bicyclist”, although the Waymo dataset includes a similar label “cyclist”, we have excluded all annotations of this category as described in Sec. 4.1 of our main paper. Moreover, given that the Waymo dataset consists largely of videos, resulting in numerous similar images, we implemented a sampling strategy. Each video was subsampled with a frame rate of 20, reducing the total number of images from 790,405 to 39,750 (denoted as 39k). We used the same hyperparameters for BLIP-2 and CLIP in our Data Feeder and Model Updater as were used for the Mapillary and nuImages datasets, respectively, for image querying and pseudo-labeling.

As indicated in Table 8, incorporating the Waymo dataset into our Data Feeder for image querying resulted in a 1.9% AP improvement in detecting novel categories, compared to using only the Mapillary or nuImages datasets. Moreover, by adding more unlabeled images from Waymo and the full BDD100k dataset, we can boost the performance to 19.8% AP, approaching the fully-supervised result of 24.1% AP. Note that the cost of AIDE is only $2.4 with 19.8% AP. This significant improvement demonstrates that our AIDE can effectively scale up with an expanded image search space.

D. More Analysis

D.1. Ablation Study of the Scaling Ratio for CLIP filtering

As discussed and illustrated in Sec. 3.3.1 and Fig. 5 of our main paper, we increase the size of the pseudo-box used to crop the image before submitting the cropped image patch for zero-shot classification (ZSC). We present an ablation study of the scaling ratio, ranging from 1.0 to 2.0, where a scaling ratio of 1.0 signifies using the pseudo-box dimensions as they are to crop the image patch. As Table 9 demonstrates, the performance of novel categories improves as the scaling ratio increases, reaching a plateau when the scaling ratio is 1.75. This trend is expected since a substantially rescaled box might include excessive background context, potentially distracting the ZSC process of CLIP. Therefore, we use a scaling ratio of 1.75 for all our experiments.

D.2. Analyzing the Data Feeder and Model Updater on Improving the Quality of Pseudo-labeling

We analyze the impact of our Data Feeder and Model Updater on improving the quality of pseudo-labels. As outlined in Section 3.2 of our main paper, our Data Feeder is designed to query images relevant to novel categories from the image pool. This process helps eliminate trivial or unrelated images during training, thereby reducing training time and enhancing performance. Moreover, our two-stage pseudo-labeling in our Model Updater will filter out raw pseudo-labels generated by OWL-v2.

To establish a baseline for comparison, we initially used OWL-v2 to perform inference on the entire image pool, i.e., Mapillary or nuImages datasets for each novel category. We measured the precision of the pseudo-labels for novel categories against the ground-truth labels in each dataset, considering a pseudo-label as a true positive if it achieved an Intersection over Union (IoU) greater than 0.5 with the

Table 9. Ablation study of the scaling ratio of the pseudo-box to crop the image patch for CLIP filtering.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Category Name</th>
<th>Scaling Ratio 1</th>
<th>1.25</th>
<th>1.50</th>
<th>1.75</th>
<th>2.00</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mapillary</td>
<td>motorcyclist</td>
<td>3.6</td>
<td>6.1</td>
<td>7.6</td>
<td>8.4</td>
<td>8.9</td>
</tr>
<tr>
<td>Mapillary</td>
<td>bicyclist</td>
<td>9.3</td>
<td>10.7</td>
<td>12.0</td>
<td>11.9</td>
<td>12.2</td>
</tr>
<tr>
<td>nuImages</td>
<td>const. vehicle</td>
<td>5.8</td>
<td>5.0</td>
<td>4.8</td>
<td>5.7</td>
<td>5.4</td>
</tr>
<tr>
<td>nuImages</td>
<td>trailer</td>
<td>2.1</td>
<td>2.1</td>
<td>3.2</td>
<td>3.6</td>
<td>3.6</td>
</tr>
<tr>
<td>nuImages</td>
<td>traffic cone</td>
<td>28.6</td>
<td>30.2</td>
<td>28.6</td>
<td>30.7</td>
<td>29.2</td>
</tr>
<tr>
<td>Average AP(%)</td>
<td></td>
<td>9.9</td>
<td>10.8</td>
<td>11.2</td>
<td>12.0</td>
<td>11.8</td>
</tr>
</tbody>
</table>
E. Limitations

Our work proposed the first automated data engine, AIDE, based on VLMs and LLMs for autonomous driving. However, there are still limitations in our work. As AIDE is extensively integrated with VLMs and LLMs, the hallucination of VLMs and LLMs may have negative impacts on our Issue Finder and Verification. Although the dense captioning model in our Issue Finder can automatically identify the novel category with high precision, it may also potentially hallucinate novel categories that are not present in the image. On the other hand, although our Verification can generate diverse scene descriptions for evaluating our detector, it may also hallucinate scenarios that do not exist in the image pool.

Generally, we believe that these concerns will be alleviated with the advancement of VLMs and LLMs in the future. Additionally, using a large image pool for text-based retrieval in Data Feeder can help mitigate these concerns. Despite the effectiveness of AIDE for a safety-critical system, some human oversight is always recommended.

F. More Experimental Details

In this section, we provide more experimental details for our AIDE and also the comparison methods. For all approaches, including supervised training, semi-supervised learning, and AIDE, we begin with the same Faster RCNN model pretrained by the same six AV datasets then proceed to conduct our experiments. For the Unbiased Teacher-v1 [5], we use the official implementation and adhere to the same training settings. Both Supervised Training and AIDE are trained for 3000 iterations, using SGD optimization with a batch size of 4, a learning rate of 5e-4, and weight decay set at 1e-4 across all experiments. The Unbiased Teacher-v1 [5] requires a warm-up stage to pre-train a teacher model, so we allocate an additional 1000 iterations, totaling 4000 iterations, for training this method. All other training hyperparameters for the Unbiased Teacher-v1 [5] remain consistent with those used for Supervised Training and AIDE. For the image-text matching in Data Feeder, we leverage the ‘pretrain’ configuration to initialize the BLIP-2 model, which is exactly based on the official BLIP-2 GitHub Repo. The VLMs we used are allowed for commercial usage (i.e., Otter/CLIP/BLIP-2). ChatGPT can be replaced by open-source LLMs like Llama2 [78], whereas the cost of ChatGPT is negligible (less than $0.01).

F.1. Model Hyperparameters for Data Feeder and Model Updater

In this section, we detail the model hyperparameter selection for our Data Feeder and Model Updater. Within our Data Feeder, we utilize BLIP-2 to query images relevant to each novel category. This is achieved by measuring the cosine similarity score between the text and image embeddings. Subsequently, all images are ranked based on their cosine similarity score (denoted as the BLIP-2 score), and the top-ranked images are selected by thresholding the BLIP-2 score. We have set the BLIP-2 score threshold at 0.6 for all novel categories. This threshold is chosen to ensure that our Data Feeder retrieves at least 1% of the images from the image pool (comprising either Mapillary or nuImages datasets) for each novel category. Such a threshold guarantees that we have a sufficient number of images for pseudo-labeling in Model Updater.

Second, in our Model Updater, given that the number of relevant images has been significantly reduced following the BLIP-2 querying process (for example, only 550 images for “motorcyclist”), we opt for a CLIP score threshold, specifically 0.1, for our two-stage pseudo-labeling to prevent excessive filtering out of too many potential pseudo-labels. As demonstrated in Section D.2 and Table 10, even

Table 10. Evaluate the quality of the pseudo-labels of novel categories generated by OWL-v2 without any post-processing, filtered by the Data Feeder with BLIP-2, and further filtered by Model Updater. We measure the precision (%) by comparing the pseudo labels with ground-truth labels for each novel category. Given a pseudo-label, we treat it as a true positive if it has an IoU larger than 0.5 with the ground-truth label, otherwise it is a false positive.

<table>
<thead>
<tr>
<th>Category</th>
<th>OWL-v2 [11]</th>
<th>w/ Data Feeder</th>
<th>w/ Model Updater</th>
</tr>
</thead>
<tbody>
<tr>
<td>motorcyclist</td>
<td>11.1</td>
<td>19.3</td>
<td>47.2</td>
</tr>
<tr>
<td>bicyclist</td>
<td>5.3</td>
<td>7.6</td>
<td>33.8</td>
</tr>
<tr>
<td>const. vehicle</td>
<td>11.3</td>
<td>12.8</td>
<td>16.5</td>
</tr>
<tr>
<td>trailer</td>
<td>10.9</td>
<td>12.1</td>
<td>38.2</td>
</tr>
<tr>
<td>traffic cone</td>
<td>68.3</td>
<td>76.9</td>
<td>92.9</td>
</tr>
<tr>
<td>Average AP(%)</td>
<td>21.4</td>
<td>25.7</td>
<td>45.7</td>
</tr>
</tbody>
</table>

Table 10 shows that compared to the raw pseudo-labels generated by OWL-v2, our Data Feeder alone improved the average precision of novel categories by 4.3%. Furthermore, when combined with our Model Updater, the average precision was enhanced to 45.7%, which is a 24.3% improvement over the raw pseudo-labels from OWL-v2. This significant improvement underscores the effectiveness of our AIDE in fine-tuning OWL-v2, surpassing the self-training method proposed by OWL-v2 in [11], as our AIDE provides substantially better quality pseudo-labels.
with such a CLIP score threshold, we can still markedly enhance the quality of pseudo-labels compared to using only the Data Feeder to filter OWL-v/2’s pseudo-labels. For filtering pseudo-labels of known categories, we set the confidence score threshold at 0.6. This threshold significantly reduces the number of pseudo-labels for each known category, helping to balance it with the number of pseudo-labels for novel categories. Such a balance is crucial in mitigating forgetting while simultaneously boosting performance for novel categories.

F.2. Experimental Details for fine-tuning OWL-v2 with AIDE

For the experiment of fine-tuning the OWL-v2 [11] with AIDE, we leverage the official model released by the author. We opted to use the Hugging Face Transformers library to fine-tune the OWL-v2 as it provides a consistent codebase for both inferring and training OWL-v2 in PyTorch. Notably, the OWL-v2 [11] was self-training on the OWL-ViT [4] on a web-scale dataset, i.e., WebLI [43], and the fine-tuning learning rate is 2e-6. To enable effective continual fine-tuning with AIDE, we set the initial learning rate as 1e-7. This setting is intended to prevent dramatic changes in the weights of OWL-v2, thereby avoiding catastrophic forgetting while still allowing the model to learn novel categories using AIDE effectively. We utilize the same training hyperparameters from the self-training recipe of OWL-v2 [11] to conduct self-training of OWL-v2 on AV datasets in Section B, ensuring a fair comparison.

F.3. Details for our Verification

As mentioned in our main paper Sec. 3.4, we leverage LLM, i.e., ChatGPT [12], to generate diverse scene descriptions to evaluate the updated detector from our Model Updater. The prompt template we use for this purpose is illustrated in Figure 9. Further, we have detailed the training process triggered by Verification in Section B. We use the same training and model hyperparameters for our continual training in Model Updater when conducting the training triggered by Verification.

G. More Visualizations

G.1. Predictions with Different Methods

We present additional visualization results in Figures 10, 11, and 12. These visualizations reveal that the Semi-Supervised Learning (Semi-SL) method tends to overfit to novel categories, resulting in numerous false positive predictions. Furthermore, the Semi-SL method struggles to detect known categories, indicating an issue with catastrophic forgetting. In contrast, the state-of-the-art Open-Vocabulary Object Detection (OVOD) method, specifically OWL-v2, also produces many false positives for both novel and known categories. However, compared to both the Semi-SL and OVOD methods, AIDE demonstrates superior performance in accurately detecting both novel and known categories.

G.2. Prediction after updating our model by Verification

In Figure 13, we present additional visualizations to Fig. 7 in our main paper to demonstrate that an extra round of training, initiated by Verification, further reduces both missed and incorrect detections of novel categories. These visualizations illustrate the effectiveness of the additional training round in enhancing the accuracy and reliability of our detection system for these novel categories.

H. Discuss about de-duplication process for video data

The nuImages dataset contains 13 frames per scene, spaced 0.5 seconds apart. Currently, we directly use all unlabeled images of nuImages dataset for Data Feeder to query without using any de-duplication process in our main paper. In practice, as the dataset gets larger or with a higher frame rate, de-duplication could further improve the data diversity for querying in Data Feeder and may potentially improve the performance of AIDE, and we leave this for future study.
I. Comparison between Verification and Active Learning alternatives

We compare our approach, “LLM description+BLIP-2” for Verification, with two Active Learning (AL) baselines. The first one is to verify the boxes predicted as the novel target class by the detector but with the highest classification entropy. The second one is to perform verification on randomly sampled boxes predicted as the novel target class by the detector. For both AL baselines, we use them to verify 10 images, the same as what we have done in Sec. 4.3.4 of our main paper. The two AL baselines only achieve 13.1% and 12.7% AP on novel classes, respectively. This is inferior to our approach (14.2% AP) which uses VLM/LLM to identify diverse AV scenarios for verification.

J. Discussion for the real-cost of supervised and semi-supervised methods

In our main paper Fig. 1, Tab. 1, and Tab. 2, we only measure the “Labeling and Training” cost for the supervised/semi-supervised methods. In fact, the real cost for the supervised/semi-supervised method is not just labeling images but also includes searching over the large data pool to find relevant images to label. For instance, an annotator needs to examine 874 images on average to find 50 images for a selected novel class, costing $43.7 for supervised/semi-supervised methods, assuming it costs 10 seconds per image to inspect for novel classes, which corresponds to $0.05 at $18 per hour. Therefore, AIDE is more practical than supervised/semi-supervised methods for car companies as we automate data querying in Data Feeder to largely reduce the total cost.
Figure 10. Visualization of the detection results under different methods. We treat a box prediction as true positive if it has an IoU larger than 0.5 with the ground-truth box. The true positive predictions are in green color, while the false positive predictions are in red color. **Top-left**: Semi-supervised Learning (Semi-SL) method, i.e., Unbiased Teacher-v1 [5]. **Top-right**: Open-vocabulary object detection (OVOD) method, i.e., OWL-v2 [11]. **Bottom-left**: AIDE. **Bottom-right**: ground-truth.
Figure 11. Visualization of the detection results under different methods. We treat a box prediction as true positive if it has an IoU larger than 0.5 with the ground-truth box. The true positive predictions are in green color, while the false positive predictions are in red color. Top-left: Semi-supervised Learning (Semi-SL) method, i.e., Unbiased Teacher-v1 [5]. Top-right: Open-vocabulary object detection (OVOD) method, i.e., OWL-v2 [11]. Bottom-left: AIDE. Bottom-right: ground-truth.
Figure 12. Visualization of the detection results under different methods. We treat a box prediction as true positive if it has an IoU larger than 0.5 with the ground-truth box. The true positive predictions are in green color, while the false positive predictions are in red color. **Top-left:** Semi-supervised Learning (Semi-SL) method, i.e., Unbiased Teacher-v1 [5]. **Top-right:** Open-vocabulary object detection (OVOD) method, i.e., OWL-v2 [11]. **Bottom-left:** AIDE. **Bottom-right:** ground-truth. Note that some original annotations in Mapillary are not correct. For instance, for the image of “GT” in the second row, the human on the bicycle should be labeled as “bicyclist” while the original label is “person”.
**LLM output:** A daytime image depicting a vibrant red **motorcyclist** moving on a busy city road.

**Queried image:** train

**Verification:** The red **motorcyclist** is not detected ✗

**Val (before 2nd round training):**

**Verification:** The red **motorcyclist** is not detected ✗

**Val (after 2nd round training):**

**Verification:** The red **motorcyclist** is detected ✔

**LLM output:** A dusk scene showing a black motorcyclist wearing a black helmet, maneuvering through heavy traffic on a crowded urban street.

**Queried image:** train

**Verification:** The **motorcyclist** is not detected ✗

**Val (before 2nd round training):**

**Verification:** The **motorcyclist** is not detected ✗

**Val (after 2nd round training):**

**Verification:** The **motorcyclist** is detected ✔

Figure 13. More visualizations on our **Verification.** **Left:** In the queried image from the training set for verification, the model is not predicting the motorcyclist. **Middle:** Similarly on the queried image from the validation set, the model is not predicting the motorcyclist. **Right:** After updating the model again, our model can successfully predict the motorcyclist.
<table>
<thead>
<tr>
<th></th>
<th>Cityscapes</th>
<th>KITTI</th>
<th>BDD100k</th>
<th>nuImages</th>
<th>Mapillary</th>
<th>Waymo</th>
</tr>
</thead>
<tbody>
<tr>
<td># Classes</td>
<td>8</td>
<td>3</td>
<td>10</td>
<td>10</td>
<td>37</td>
<td>3</td>
</tr>
<tr>
<td>Cumulative # Classes</td>
<td>8</td>
<td>8</td>
<td>10</td>
<td>10</td>
<td>37</td>
<td>46</td>
</tr>
<tr>
<td># Images</td>
<td>2,975</td>
<td>6,859</td>
<td>69,863</td>
<td>67,279</td>
<td>18,000</td>
<td>790,405</td>
</tr>
</tbody>
</table>

Vehicle

- car
- truck
- bus
- train
- motorcycle
- bicycle

Human

- person
- rider
- pedestrian
- cyclist

Traffic Objects

- traffic cone
- traffic light
- traffic sign
- barrier
- traffic sign frame

Other Objects

- bird
- ground animal
- crosswalk plain
- lane marking crosswalk
- banner
- bench
- bike rack
- billboard
- catch basin
- cctv camera
- fire hydrant
- junction box
- mailbox
- manhole
- phone booth
- trash can

Table 11. The statistics and label space of the six AV datasets, i.e., Cityscapes [47], KITTI [48], BDD100k [49], nuImages [45], Mapillary [50], and Waymo [46]. There are 46 categories in total after combining the label spaces. To simulate the novel categories and ensure that the selected categories are meaningful and crucial for AV in the street, we choose 5 categories as novel categories: “motorcyclist” and “bicyclist” from Mapillary, “construction vehicle”, “trailer”, and “traffic cone” from nuImages. The rest 41 categories are set as known. We remove all the annotations for these categories in our joint datasets and also remove the related categories with similar semantic meanings, e.g., “bicyclist” vs “cyclist”, “rider” vs “motorcyclist”.