# Effective Adaptation in Multi-Task Co-Training for Unified Autonomous Driving

**Xiwen Liang**[1*], **Yangxin Wu**[1*], **Jianhua Han**[2], **Hang Xu**[2], **Chunjing Xu**[2], **Xiaodan Liang**[1†]
[1]Shenzhen Campus of Sun Yat-Sen University, [2]Huawei Noah's Ark Lab
{liangxw29@mail2, wuyx29@mail2, liangxd9@mail}.sysu.edu.cn,
{hanjianhua4, xu.hang, xuchunjing}@huawei.com

## Abstract

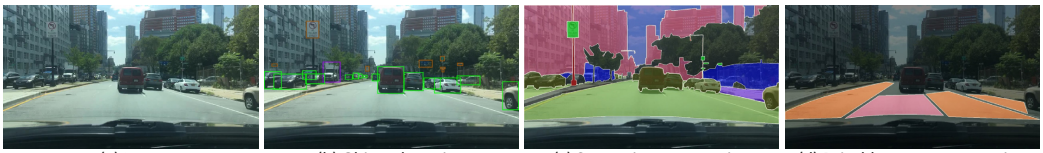
Aiming towards a holistic understanding of multiple downstream tasks simultaneously, there is a need for extracting features with better transferability. Though many latest self-supervised pre-training methods have achieved impressive performance on various vision tasks under the prevailing *pretrain-finetune* paradigm, their generalization capacity to multi-task learning scenarios is yet to be explored. In this paper, we extensively investigate the transfer performance of various types of self-supervised methods, e.g., MoCo and SimCLR, on three downstream tasks, including semantic segmentation, drivable area segmentation, and traffic object detection, on the large-scale driving dataset BDD100K. We surprisingly find that their performances are sub-optimal or even lag far behind the single-task baseline, which may be due to the distinctions of training objectives and architectural design lied in the *pretrain-finetune* paradigm. To overcome this dilemma as well as avoid redesigning the resource-intensive pre-training stage, we propose a simple yet effective *pretrain-adapt-finetune* paradigm for general multi-task training, where the off-the-shelf pretrained models can be effectively adapted without increasing the training overhead. During the *adapt* stage, we utilize learnable multi-scale adapters to dynamically adjust the pretrained model weights supervised by multi-task objectives while leaving the pretrained knowledge untouched. Furthermore, we regard the vision-language pre-training model CLIP as a strong complement to the *pretrain-adapt-finetune* paradigm and propose a novel adapter named LV-Adapter, which incorporates language priors in the multi-task model via task-specific prompting and alignment between visual and textual features. Our experiments demonstrate that the *adapt* stage significantly improves the overall performance of those off-the-shelf pretrained models and the contextual features generated by LV-Adapter are of general benefits for downstream tasks.

## 1 Introduction

Multi-task learning is a long-studied problem in computer vision and has become an emerging paradigm in autonomous driving [22, 54, 21]. An autonomous vehicle may need to concurrently perform a wide range of perception tasks, e.g., locating the pedestrians and cars, deciding road affordability for driving, as well as detecting the lanes to precipitate a driving action. The joint learning of multiple tasks can not only reduce the training and inference time, but also act as a regularizer that enforces the learning of generalizable representations [6, 22]. Apart from efficiency and simplicity, an all-in-one architecture is argued to have the potential to enhance the robustness of the driving system by implicitly learning the synergy between heterogeneous tasks [22, 49, 44].

[*]Equal contribution.
[†]Corresponding author.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).



(a) Input (b) Object detection (c) Semantic segmentation (d) Drivable area segmentation

Figure 1: Our multi-task model takes (a) an RGB image as input and tackles (b) traffic object detection, (c) semantic segmentation, and (d) drivable area segmentation simultaneously.

In the pursuit of the merits of multi-task learning, it is crucial to obtain universal features with great transferability. Recently, many concurrent self-supervised pre-training methods have shown great potential when transferred to different types of vision tasks [51, 53, 48] under the *pretrain-finetune* paradigm. Despite the impressive performance, their transferability to multi-task learning scenarios is yet to be explored. We argue that the heterogeneity of tasks will introduce a multitude of challenges to train a unified model and it is often not the case that multi-task learning is of universal benefit. Firstly, the widely adopted *pretrain-finetune* paradigm may lead to degraded performance in multi-task learning due to the misalignment of objectives between pre-training and fine-tuning [14, 19] since most supervised and self-supervised methods are highly specialized for a specific type of objective or task [3, 4, 51, 53]. Secondly, the performance of multi-task learning relies on many non-trivial factors including model architectures, data augmentations, hyperparameters, convergence properties, etc [40, 43]. The specialized techniques catered for a specific type of architecture or task are largely not applicable to a universal architecture since they are prone to fail during generalization. Moreover, due to the labor-intensive process of data annotation, it is hard to collect complete annotations of different granularities for all tasks, which further complicates the situation.

In this paper, we focus on heterogeneous multi-task learning on partially labeled data (HMPL) under the realistic scenario of autonomous driving. We first delve into the representation learning stage of HMPL to reveal the performance degradation of different pre-training methods. We thoroughly examine a wide range of pre-training methods including supervised pre-training (pretrained on ImageNet [37] ), classification-oriented methods (e.g., SimCLR [3], MoCo [18]), detection-oriented methods (e.g., DetCo [51]), segmentation-oriented methods (e.g., DenseCL [48]), and vision-language pre-training methods (e.g., CLIP [33]) on three fine-grained tasks on the large-scale driving dataset BDD100K [56], i.e., object detection, semantic segmentation, and drivable area segmentation. Surprisingly, the performance of these methods varies greatly under the same training protocol, especially on the dense prediction task. For example, MoCov2 [4] only gets 12.5 mIoU in semantic segmentation and segmentation-oriented models like DenseCL [48] and DetCo [51] perform unsatisfactorily on dense prediction tasks, which suggests that the misalignment lied in the *pretrain-finetune* paradigm can lead to prominent performance degradation and there exists much room for improvement.

Given that 'universal' pre-training is still unsolved and redesigning the resource-intensive pre-training scheme comes with great computation overhead, we aim to develop a general approach that can fully harness the knowledge from the off-the-shelf pretrained models and make them amenable to multi-task scenarios via efficient adaptation. We draw inspiration from the recent progress of prompt-based learning in NLP [27, 59, 33, 12], where the language model pretrained on massive amounts of raw text can be adapted to new scenarios with high efficiency by introducing hand-crafted or learnable prompts. Following this philosophy, we propose a simple yet effective *pretrain-adapt-finetune* paradigm for multi-task transfer learning as a substitute for the dominating *pretrain-finetune* paradigm in computer vision. Concretely, during the *adapt* stage, learnable multi-scale adapters with a small amount of parameters are tuned with frozen random initialized task-specific heads to dynamically adapt the knowledge from the pretrained models under the multi-task objectives. Albeit simple, we show that the *adapt* stage mitigates the gap between pre-training and fine-tuning and significantly improves the overall performance across different pretrained models while being very effective and introducing no extra training overhead.

Apart from the supervised and self-supervised pre-training, Contrastive Language-Image Pre-training (CLIP) [33] manages to learn high-quality visual representation from an enormous amount of noisy image-text pairs. CLIP achieves breakthrough performance in zero-shot image recognition, which indicates that it might have great potential for improving the generality and robustness of multi-task learning. To this end, we propose LV-Adapter to further excavate the linguistic knowledge from CLIP

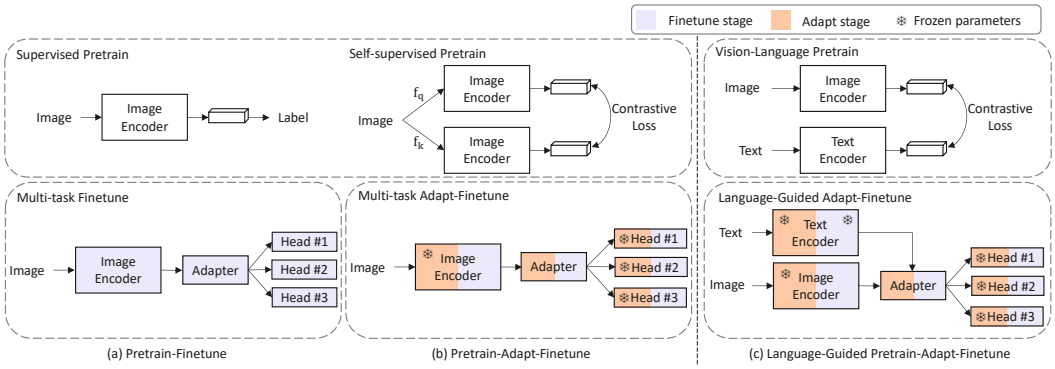



Figure 2: Comparisons of the conventional *pretrain-finetune* paradigm and our proposed *pretrain-adapt-finetune* paradigm. The language-guided *pretrain-adapt-finetune* paradigm further incorporates language priors into multiple downstream tasks.

and enhance the visual representations in the multi-task scenarios. To maximize the correspondence between the dense features and the class-specific concepts, we first learn task-specific prompts in an end-to-end manner. Then we model the language-to-vision adaptation function based on the cross-attention mechanism [47] to incorporate language priors in visual features.

We experimentally verify that our proposed *pretrain-adapt-finetune* paradigm significantly improves the overall performance of different off-the-shelf pre-training methods. For example, it obtains an absolute improvement by over 40% in semantic segmentation on MoCo-v1/v2, and DenseCL [18, 4, 48], and its superior performance does not rely on extra training costs or sophisticated architectural design. Furthermore, we showcase that our proposed LV-Adapter can serve as an effective complementary approach for multi-task learning. Especially, we experiment on three settings that correspond to different levels of annotation quantity and observe that LV-Adapter brings consistent performance gains on three heterogeneous tasks simultaneously. We hope our work can shed light on the heterogeneous multi-task learning by showing that effective adaptation is of great essence to mitigate the deficiency in the dominating *pretrain-finetune* paradigm.

## 2 Related Work

**Multi-Task Learning** Multi-task learning aims to improve efficiency and accuracy through shared information among multiple tasks. MultiNet [45] joints classification, detection and semantic segmentation via a unified architecture. Eigen *et al.* [10] address depth prediction, surface normal estimation, and semantic labeling. YOLOP [49] leverages CSPDarknet as the backbone and adopts three specific decoders to solve object detection, drivable area segmentation, and lane detection respectively. Standley *et al.* [40] and Fifty *et al.* [11] propose to identify proper task groupings for multi-task training instead of naively training all tasks together. In this paper, we focus on developing general and effective approaches for multi-task transfer learning in autonomous driving scenarios. We provide more discussions with several related works in the Appendix.

**Pre-training Methods** A dominating paradigm in computer vision is to pretrain on a large scale of data, e.g., ImageNet [37], then finetune on target tasks with usually less training data. Recently, researchers are interested in learning visual representations without human supervision. We roughly divided them into three classes, namely, classification-oriented, detection-oriented, and segmentation-oriented methods. Classification-oriented methods typically rely on contrastive learning and online clustering, e.g., SimCLR [3], MoCo [18, 4], and BYOL [16]. Detection-oriented methods like DetCo [51] are specially designed for object detection by conducting contrastive learning between the global image and local patches. Segmentation-oriented methods like PixPro [53] further work on pixel-level correspondence to benefit dense prediction downstream tasks. We show that these methods are sub-optimal for multi-task learning, and we focus on improving the performance of these off-the-shelf methods instead of redesigning the computation-intensive pre-training stage.

**Prompting Learning** Natural language prompting freezes large-scale pretrained language models and reformulates text input with example or learnable prompts to bridge the gap between pre-training

Table 1: Comparisons of popular multi-task training methods and our proposed LV-Adapter under the Disjoint-normal setting.

| Method | mIoU (SS) | mIoU (DA) | mAP | AP50 | AP75 |
|---|---|---|---|---|---|
| Zeroing loss [50] | 59.4 | 83.3 | 23.0 | 46.0 | 19.8 |
| Uniform sampler [25] | 59.9 | 83.2 | 24.3 | 47.0 | 21.8 |
| Weighted sampler [25] | 59.8 | 83.2 | 24.2 | 46.9 | 21.4 |
| Round-robin [25] | 60.7 | 83.1 | 24.2 | 47.0 | 21.5 |
| Self-training [14] | 60.3 | 83.1 | 24.9 | 48.1 | 22.2 |
| **LV-Adapter (Ours)** | **62.2** | **83.7** | **26.4** | **50.5** | **23.7** |

and model tuning efficiently. GPT-3 [1] first introduces in-context learning, prepending hard text prompts for every task before the input. Then many works based on such discrete text prompts have emerged, i.e., designing prompts manually [41, 38] and searching trigger words [39, 13]. Furthermore, recent methods inject learnable soft prompts [24, 29, 57] into the embedding space of the model and achieve impressive performance. In the field of computer vision, CLIP [33] trains on millions of vision-language pairs and demonstrates effective zero-shot classification. CoOp [59], CPT [55], and CLIP-Adapter [12] further tunes fixed CLIP with soft prompts by few-shot supervisions.

# 3 Empirical Study

## 3.1 Heterogeneous Multi-Task Setup

We concentrate on three challenging heterogeneous tasks in autonomous driving, i.e., traffic object detection, semantic segmentation, and drivable area segmentation, on large-scale driving dataset BDD100K [56]. We showcase the input and output of our model in Figure 1. We select the state-of-the-art detector and segmentation decoder head to build up our model. To be specific, we follow the hard-parameter sharing scheme [6] where each task shares the same backbone and has its task-specific head. We outline our model architecture in Figure 2 and elaborate each component in the following.

**Backbone & Neck.** We adopt the classic ResNet [20] as backbone and FPN [26] as neck to generate multi-scale features. The FPN neck [26] constructs pyramidal features, namely, $\{P_2, P_3, P_4, P_5\}$, which have strides of $\{4, 8, 16, 32\}$ pixels w.r.t the input image and fixed 256 channels.

**Segmentation Head.** We implement the decoder for segmentation based on MaskFormer [5]. MaskFormer formulates the segmentation task as a mask classification problem and its head is composed of a fully convolutional pixel decoder and a transformer decoder module. The pixel decoder takes $\{P_2, P_3, P_4, P_5\}$ as input and gradually upsamples the features to produce high resolution per-pixel embeddings $\varepsilon_{\text{pixel}}$. The transformer decoder module utilizes a fixed set of queries to attend to image features and produces mask embeddings $\varepsilon_{\text{mask}}$. Then $\varepsilon_{\text{pixel}}$ and $\varepsilon_{\text{mask}}$ is multiplied together to generate prediction masks. We refer the reader to [5] for more details.

**Detection Head.** We implement our detector based on Sparse R-CNN [42]. The detection pipeline constructs a fixed set of learnable proposal boxes (e.g., 300) which serve as region proposals and are fed into a series of heads for prediction. The dynamic instance interactive head dynamically generates the weights of a convolution filter that attends to the bins in the region proposals and predicts the location and category of object in a cascade manner.

We introduce three settings, i.e., the Disjoint-normal, Disjoint-balance, and Full-setting, corresponding to different levels of annotation quantity in our experiments. For clarity, we defer the detailed descriptions of three settings to Section 5.1. For consistency, we conduct evaluations on the same validation set of BDD100K for different settings above. We mainly work under the Disjoint-normal setting for comparisons between the performance of different pre-training methods as well as ablation studies. And we further verify the efficacy of our proposed methods on the Disjoint-balance and the Full-setting in Section 5.2.

Table 2: Comparisons of different paradigms under the Disjoint-normal setting with ResNet-50 backbone. Orange color indicates the results of our proposed *pretrain-adapt-finetune* paradigm, while others are results of conventional *pretrain-finetune* paradigm.

| | | Semantic Seg. | | Drivable Seg. | | Object Detection | | |
|---|---|---|---|---|---|---|---|---|
| Type | Model | mIoU | pACC | mIoU | pACC | mAP | AP50 | AP75 |
| Classification-oriented | MoCo-v1 [18] | 17.8 | 48.6 | 70.8 | 92.0 | 25.8 | 49.5 | 23.1 |
| | | 59.2 | 93.2 | 83.6 | 96.9 | 25.9 | 50.0 | 23.0 |
| | MoCo-v2 [4] | 10.3 | 19.8 | 73.4 | 93.5 | 26.0 | 50.1 | 23.2 |
| | | 61.2 | 93.4 | 83.8 | 96.9 | 26.1 | 50.4 | 23.4 |
| | SimCLR [3] | 60.3 | 93.3 | 83.5 | 96.8 | 25.4 | 48.9 | 22.5 |
| | | 60.1 | 93.2 | 83.5 | 96.8 | 25.2 | 48.9 | 22.3 |
| | SwAV [2] | 45.9 | 71.1 | 82.0 | 96.5 | 25.6 | 49.1 | 23.0 |
| | | 61.1 | 93.3 | 83.1 | 96.7 | 25.6 | 49.3 | 23.1 |
| | BYOL [16] | 59.2 | 90.2 | 75.6 | 93.9 | 25.9 | 49.8 | 23.4 |
| | | 61.7 | 93.4 | 83.5 | 96.8 | 25.7 | 49.4 | 23.1 |
| Detection-oriented | DetCo [51] | 38.1 | 58.5 | 83.2 | 96.7 | 25.9 | 49.7 | 22.9 |
| | | 61.0 | 93.4 | 83.7 | 96.9 | 26.2 | 50.3 | 23.3 |
| Segmentation-oriented | DenseCL [48] | 20.0 | 40.0 | 73.7 | 93.7 | 26.1 | 50.3 | 23.5 |
| | | 60.7 | 93.3 | 83.9 | 96.9 | 26.3 | 50.3 | 23.7 |
| Vision-language | CLIP [33] | 54.5 | 91.1 | 74.1 | 93.1 | 26.5 | 50.7 | 23.8 |
| | | 61.0 | 93.2 | 83.4 | 96.8 | 26.3 | 50.5 | 23.5 |

### 3.2 Multi-Task Learning Baselines

**Self-training** Based on the available annotations for each task, we trained three single-task teacher models on the labeled data and use them to generate pseudo labels on unlabeled data following [52, 14]. For single-task teacher models, we adopt the same architecture and training schedules of Sparse R-CNN [42] and MaskFormer [5]. For object detection, we use a score threshold of 0.5 to select pseudo box labels. For segmentation, we use the score threshold of 0.3 to select segmentation masks. Pixels with lower scores are assigned with the ignore label. Thereafter, every image has annotations for all tasks and we train a multi-task student model with a weighted sum of all objectives for each task:

$$\mathcal{L}_{total} = \alpha_{det}\mathcal{L}_{det} + \alpha_{sem}\mathcal{L}_{sem} + \alpha_{driv}\mathcal{L}_{driv}, \tag{1}$$

where $\mathcal{L}_{det}, \mathcal{L}_{sem}, \mathcal{L}_{driv}$ are losses for object detection, semantic segmentation, and drivable area segmentation, respectively. [1] We include more details in Section 5.1.

**Zeroing loss** Some works [22, 50] simply zero losses for a particular task if the input image does not have the corresponding annotation.

**Batch-level Round-Robin** In this scheme [56, 25], each batch consists of samples that have the same type of annotations for a task and different tasks are repeated in a fixed order during training.

**Task Scheduling** This is a stochastic schedule that the task for each SGD update step is sampled from a distribution [25]. We mainly consider a uniform task sampling schedule that sampled from a uniform distribution, and a weighted task sampling schedule where the sampling weight for a task is proportional to the number of labeled images and training epochs for this task.

**Performance Comparison** As can be seen in Table 1 and Table 3, these methods surpass single-task teacher model by a clear margin, which indicates that the model benefits from multi-task learning. Among these baselines, self-training achieves satisfactory performance while being conceptually simple, thus we adopt it for the rest experiments in this paper.

[1] We conduct grid search in the range of [0.1, 1.0] with a step size of 0.1 to find the optimal loss weights setting and $\alpha_{det}, \alpha_{sem}, \alpha_{driv}$ are set to $1.0, 0.7, 0.7$, respectively. We fix the loss weights throughout the paper.

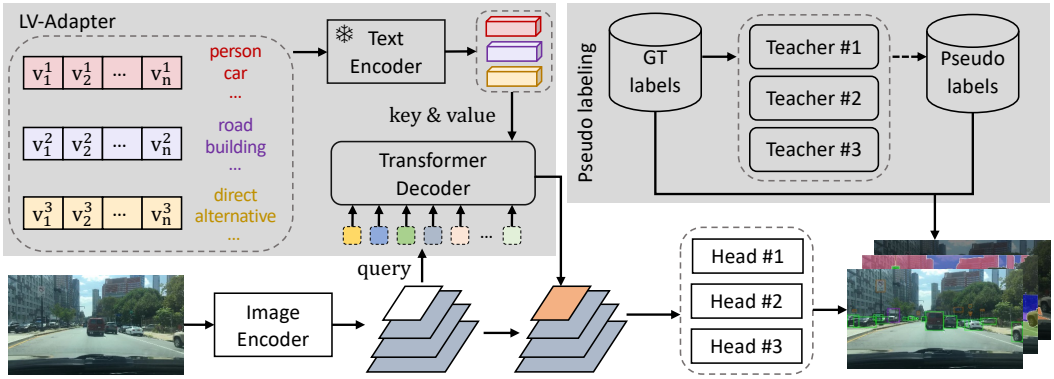



Figure 3: An overview of our proposed model. We first train three specialized teacher on labeled data to generate pseudo labels for each task. The multi-task model is then trained on both ground-truth and pseudo labels under the language-guided *pretrain-adapt-finetune* paradigm.

### 3.3 Pretrain-finetune for Multi-Task Learning

In this section, we adopt the identical training protocol and architecture to extensively investigate the performances of different types of pre-training methods, including task-oriented methods and vision-language pre-training, when transferred to the multi-task learning scenario. We revisit the technical details of these different schemes in the Appendix. We report the performance of the aforementioned methods in Table 2, and come to the following observations:

- Only SimCLR achieves decent performance on all three tasks and many methods encounter substantial degradation on pixel-level segmentation tasks. For semantic segmentation, MoCo-v1/v2 and DenseCL achieve the worst mIoU (degrade by 66%-83% v.s. ImageNet-pretrained). For drivable area segmentation, MoCo-v1/v2, DenseCL, and CLIP have the worst performance (degrade by nearly 12% v.s. ImageNet-pretrained).
- The pre-training paradigm does not seem to have an explicit correlation with the downstream performance, and even the task-oriented design in pre-training does not guarantee the transfer performance on the corresponding task type. For example, the segmentation-oriented DenseCL achieves sub-optimal performance on two segmentation tasks.

## 4 Effective Adaptation for Multi-Task Learning

### 4.1 Pretrain, Adapt, then Finetune

We attribute the degraded transferring performance of the state-of-the-art pre-training methods to two critical factors, i.e., optimization gap and architectural gap. Firstly, the high specialization of the pre-training methods and the heterogeneity of downstream tasks result in the optimization gap in the classic *pretrain-finetune* paradigm. To be specific, the objective for pre-training is usually a type of contrastive loss (e.g., InfoNCE [46]) while the fine-tuning stage is supervised by a weighted sum of several task-specific losses. Secondly, we notice that many models in Table 2 are pretrained with a convolutional head, e.g., Fast R-CNN detector [15] or convolution-based projection head. In contrast, we adopt the transformer-based MaskFormer head for segmentation tasks. The distinctions between convolution and transformer lie in the inductive bias (local v.s. global) and the internal representation structure [34] and we conjecture this plays a non-negligible role in transferring. We further experimentally verify this point in Appendix.

The limited transfer performance presented in Table 2 indicates that the 'universal' pre-training remains unsolved. Instead of redesigning another pre-training scheme that brings vast computation overhead, we aim to efficiently reuse the intact knowledge of these off-the-shelf pre-trained models with minimal modifications to the architecture. We draw inspiration from the recent progress of prompt-based learning, where prompt tuning emerges as a new alternative to fine-tuning. For example, P-Tuning v2 [28] matches the fine-tuning performance by only tuning learnable prompts

while freezing the large-scale language model. Following this philosophy, we propose a simple yet effective *pretrain-adapt-finetune* paradigm to mitigate the gap between the pre-training and fine-tuning stage. During the *adaptation* stage, we are given the pretrained model weights (e.g., ResNet-50) inherited from the *pre-training* stage. We aim to transform the model weights via a small amount of learnable parameters to adapt the knowledge of the pretrained weights towards multi-task scenarios. To this end, we freeze the parameters of the random initialized task-specific heads and the backbone, and tune the parameters of Feature Pyramid Network (FPN) supervised by the multi-task loss function in Equation 1. During *fine-tuning*, all parameters are activated and updated via gradient descent. We compare different schemes for multi-task learning in Figure 2. This *adaptation* stage characterizes several critical design choices: (1) This stage bridges the gap between *pre-training* and *fine-tuning* by including the pretrained weights and multi-task objectives simultaneously, and the frozen backbone prevents the pretrained weights from being spoiled before the *fine-tuning* stage. (2) Compared to the single-layer prompt/adapter in P-Tuning [23] and CLIP-Adapter [12], the FPN acts as a multi-scale adapter that is endowed with greater capacity and provides semantically stronger features for fine-grained downstream tasks. (3) Taking FPN as the adapter means that the model architecture for the *pretrain-finetune* and *pretrain-adapt-finetune* paradigm is identical and effective adaptation can be achieved without any modification to the architecture. We have experimented with more sophisticated adapters (e.g., scale-aware adapters, lightweight transformers) and observed marginally better results. Hence, we opt for the original architecture above for simplicity. We experimentally verify that our *pretrain-adapt-finetune* paradigm significantly improves the performance and stability of different kinds of pre-training methods (c.f. Section 5.2) without increasing the total training costs (c.f. Section 5.3).

### 4.2 Language-to-Vision Adapter

From the perspective of transfer learning, CLIP can serve as a complementary scheme to enhance the *pretrain-adapt-finetune* paradigm, since it can comprehend the concepts in natural language. Basically, the CLIP model excels in aligning the visual and language embeddings and some works [58, 35] claim that the textual features generated by CLIP have meaningful correspondence to the semantic regions in an image. We note that the result of CLIP in Table 2 only reuses the weights of its image encoder and discards the text encoder. Therefore, we take a step further to explicitly exploit the knowledge in the full CLIP model. We pursue to underpin the compatibility between the semantic concepts of each task and the image features and generate semantically stronger contexts for downstream tasks. The resulting model, named LV-Adapter, is outlined in Figure 3, and we elaborate on each ingredient in the followings.

The CLIP model adopts ResNet [20] and BERT [7] as the encoder for image and text, respectively. Formally, we denote the last output feature maps from ResNet stages as $\{\mathbf{x}_i\}_{i=2}^{5}$. Attentional pooling and $L_2$ normalization are applied to $\mathbf{x}_5 \in \mathbb{R}^{H_5 W_5 \times C_5}$ to produce the global image feature $\hat{\mathbf{I}}_e \in \mathbb{R}^{1 \times C_5}$ for zero-shot inference, which can be formulated as follows:

$$[\hat{\mathbf{I}}_e, \hat{\mathbf{x}_5}] = \mathrm{L2_NORM}(\mathrm{MHSA}(\mathrm{GAP}(\mathbf{x}_5) \oplus \mathbf{x}_5)), \tag{2}$$

where $\mathrm{GAP}(\cdot)$ denotes global average pooling, $\mathrm{MHSA}(\cdot)$ denotes multi-head self-attention [47], and $\oplus$ denotes concatenation operation. To extract textual features of each class, class-specific prompts are constructed via a generator function and are fed into the text encoder. We denote the normalized output features for $N$ classes as $\hat{\mathbf{T}}_e \in \mathbb{R}^{N \times C}$, which can be obtained as follow:

$$\hat{\mathbf{T}}_e = \mathrm{L2_NORM}(\mathrm{TE}(\mathrm{Gen}(\{\mathbf{n}_i\}_{i=1}^{N}))), \tag{3}$$

where $\mathrm{Gen}(\cdot)$ is the generator function of class-specific prompts, TE is the text encoder, and $\{\mathbf{n}_i\}_{i=1}^{N}$ is the class name embeddings. Though $\hat{\mathbf{T}}_e$ and $\hat{\mathbf{I}}_e$ are well aligned in the image-text contrastive pre-training, they do not preserve any spatial details and are not readily applicable for downstream vision tasks. In contrast, the multi-level features output by FPN are semantically rich in the spatial dimension but are not directly aligned in the pre-training stage. Therefore, we pursue to learn task-specific prompts and propose a Language-to-Vision adapter to enhance the pixel-class correspondence.

**Learning Task-Specific Prompts.** The original generator function of CLIP outputs hand-crafted prompts using the predefined template, e.g., "`a photo of a [CLS]`.". However, the performance is highly sensitive to the form of the template [33, 59], which may be sub-optimal when transferring to heterogeneous downstream tasks. Hence, we follow CoOp [59] to use learnable textual contexts in

Table 3: Results of single-task baselines and multi-task models with ResNet-50 backbone. SS and DA means semantic segmentation and drivable area segmentation. - indicates inapplicable.

| Setting | Method | mIoU (SS) | mIoU (DA) | mAP | AP50 | AP75 |
|---|---|---|---|---|---|---|
| Full | MaskFormer [5] | 57.1 | - | - | - | - |
| | MaskFormer [5] | - | 83.9 | - | - | - |
| | Sparse R-CNN [42] | - | - | 29.4 | 55.8 | 26.4 |
| | Self-training [14] | 61.8 | 84.4 | 30.1 | 56.6 | 27.6 |
| | **LV-Adapter (Ours)** | **63.1** | **84.9** | **31.1** | **58.2** | **28.4** |
| Disjoint-balance | MaskFormer [5] | 57.1 | - | - | - | - |
| | MaskFormer [5] | - | 78.1 | - | - | - |
| | Sparse R-CNN [42] | - | - | 18.6 | 37.8 | 15.6 |
| | Self-training [14] | 59.4 | 80.3 | 22.4 | 44.1 | 19.6 |
| | **LV-Adapter (Ours)** | **61.8** | **80.6** | **24.6** | **47.4** | **21.9** |
| Disjoint-normal | MaskFormer [5] | 57.1 | - | - | - | - |
| | MaskFormer [5] | - | 82.0 | - | - | - |
| | Sparse R-CNN [42] | - | - | 20.9 | 41.9 | 17.8 |
| | Self-training [14] | 60.3 | 83.1 | 24.9 | 48.1 | 22.2 |
| | **LV-Adapter (Ours)** | **62.2** | **83.7** | **26.4** | **50.5** | **23.7** |

prompting for each task. In spite of the inconsistent output formats of the three tasks (4-D box v.s. dense output), they both need to determine the category of boxes or pixels, and we incorporate the class names of each task in prompting. The task-specific prompting can be formulated as follows:

$$\hat{\mathcal{T}_{e,i}} = \text{L2_NORM}(\text{TE}([\mathbf{v}^t, \mathbf{n}_i]), \tag{4}$$

where $\mathbf{n}_i$ refers to the embeddings of the class name belonging to a specific task, and $\mathbf{v}^t$ is the task-specific learnable contexts and the superscript indicates the task type.

**Enhancing Pixel-Class Correspondence.** To align the textural features and the dense FPN features, we propose a Language-to-Vision Adapter to incorporate the language priors into the visual features. Formally, we denote the last feature map of $P_5$ as $\mathbf{z}_5 \in \mathbb{R}^{H_5 W_5 \times C}$, and we aim to learn an adapter function $\mathcal{A}_{L\rightarrow V}$ to generate language-aware contexts for downstream tasks. We utilize the cross-attention mechanism in Transformer decoder [47] for Language-to-Vision adaptation:

$$\mathcal{A}_{L\rightarrow V}(\hat{\mathcal{T}}_e, \mathbf{z}_5) = \text{TransDecoder}(q = \mathbf{z}_5, k = \hat{\mathcal{T}}_e, v = \hat{\mathcal{T}}_e), \tag{5}$$

where the $q, k, v$ stands for query, key, and value. A single linear fully connected layer is used to adjust the channel number of $\hat{\mathcal{T}}_e$ and we omit it in Equation 5 for simplicity. We denote the output of Equation 5 as $\tilde{\mathbf{z}_5} \in \mathbb{R}^{H_5 W_5 \times C}$. After adaptation, we simply substitute $\mathbf{z}_5$ with $\tilde{\mathbf{z}_5}$ to inject linguistic knowledge into the high-level features of FPN and leave the task-specific head design unchanged. We follow the same paradigm described in Section 4.1 when incorporating LV-Adapter: only the adapter module (i.e., LV-Adapter and FPN) is tuned during the *adaptation* stage. Moreover, we freeze the weights of the CLIP text encoder throughout the paper to preserve the language priors. We verify the efficacy of our LV-Adapter and disentangle the effect of different modules ablation in Section 5.3.

# 5 Experiments

## 5.1 Experiment Settings

In BDD100K, 70k training images are labeled for both object detection and drivable area segmentation, and only 7k training images are labeled for semantic segmentation. These two sets are not disjoint and about 3k images have complete annotations for all three tasks. As is described in [14], data annotation is one of the biggest challenges of training a multi-task model. In real-world scenarios, it is unrealistic to obtain complete types of annotations for all input images, especially in the traffic scene where fine-grained annotations are required. In this paper, we are interested in the performance of multi-task learning with different quantities of annotations. Thus, we mainly consider three scenarios that correspond to three different levels of annotation scarcity, namely, Disjoint-normal, Disjoint-balance,

and Full setting. We mainly work under the Disjoint-normal setting where the annotations of each task are disjoint, and the number of labeled images for each task is in decreasing order as follows: drivable area segmentation (20k), object detection (10k), semantic segmentation (7k). We provide the details of these three settings in the Appendix.

**Implementation Details.** The training epoch is fixed as 36, syncBN [32] is on, the learning rate at the *fine-tuning* stage is $2.5 \times 10^{-5}$, and weight decay is $1 \times 10^{-4}$. The image scale is $1280 \times (720, 600)$. No other data augmentation is used. We adopt the AdamW [30] optimizer. The warmup iteration is 1000 and the warmup factor is 0.01. For LV-Adapter, the learnable prompts are prepended to the class, and the length of prompts is 16. During the *adaptation* stage, the learning rate is set to $2.5 \times 10^{-4}$. The layer of transformer decoder in Equation 5 is 3. All experiments are conducted on servers with 8 Nvidia V100 GPU (32GB) cards and Intel Xeon Platinum 8168 CPU (2.70GHz).

### 5.2 Main Results

We present the results of our proposed paradigm in Table 2. The adapt and finetune epochs of the proposed paradigm in Table 2 are set as 6 and 30 respectively. As can be seen, with the conventional *pretrain-finetune* paradigm, the performances of the state-of-the-art self-supervised methods are unstable, especially in semantic segmentation and drivable area segmentation. For example, MoCo-v2 only achieves 10.3 mIoU on semantic segmentation. In contrast, under the *pretrain-adapt-finetune paradigm*, MoCo-v2 improves by a large margin in semantic segmentation (+50.9 in mIoU). Notably, MoCo-v1/v2 and DenseCL obtain an absolute improvement by over 40% mIoU in semantic segmentation. The results demonstrate that the *pretrain-adapt-finetune* paradigm significantly improves the overall performance across different types of pretrained models. We also implement experiments with popular convolutional heads, i.e., Faster R-CNN [36] and Semantic FCN [21], as in Appendix D.2. Results show that architecture is a core reason for the degradation. Moreover, we further showcase the efficiency of this paradigm in Section 5.3. We compare the results of our LV-Adapter to single-task baselines and the self-training baseline using the ImageNet pretrained model under three settings in Table 3. It demonstrates that LV-Adapter performs better on all metrics. Therefore, we conclude that the language-guided *pretrain-adapt-finetune* paradigm can effectively reduce the gap between the *pre-training* and *fine-tuning* stage by incorporating the linguistic knowledge to visual features.

### 5.3 Ablation Study

**Module design.** We compare the performance of different module designs in Table 4 using the pre-trained weights from CLIP. Row #1 refers to the performance of self-training with the *pretrain-finetune* paradigm using the pretrained weights of the CLIP image encoder. Row #2-#5 refer to the performance of the *pretrain-adapt-finetune* paradigm and row #3-#5 reuse the pretrained weights of CLIP text encoder. The adapt and finetune epochs for the pretrain-adapt-finetune paradigm are fixed as 1 and 5. The Prompt column refers to naive prompting without alignment between visual and textual features [35]: we compute class-specific textual features by prompting the CLIP text encoder (Equation 4) and compute the class activation maps by computing dot product between the textual features and $\mathbf{z}_5$ after normalization, and the output maps are concatenated to $\mathbf{z}_5$ and a $1 \times 1$ convolution layer is used to reduce the number of channel. The V2L column refers to the post-model module in [35], where the image context is used to prompt the language model and language-compatible class activation maps are concatenated back to $\mathbf{z}_5$. In comparison, our Language-to-Vision Adapter (L2V column) directly incorporates linguistic knowledge into visual features and does not rely on dense activation maps and is thus more lightweight. As can be seen, when retrieving knowledge from pretrained text encoder via continuous prompt, the model improves in semantic segmentation by 0.3 mIoU (#2 vs #3). When we adopt the Vision-to-Language Adapter to update the text features, the performance degrades slightly in semantic segmentation (#3 vs #4). In contrast, our proposed LV-Adapter can better extract prior knowledge from CLIP and achieve the best results (+0.4 and +0.2 in mIoU in semantic segmentation and drivable area segmentation respectively).

**Number of epochs of the *adapt* and *finetune* stage.** We compare different configurations of *adapt* and *finetune* epochs in Table 5. When adapting with fewer epochs and fixing full epochs as 36, the performance improves gradually and the best configuration is 1 epoch for *adapt* and 35 epochs for *finetune*. Since we freeze the backbone and randomly initialized heads during the *adapt* stage, the number of trainable parameters is moderate. We can also see that with fixed finetune epochs, more

Table 4: Ablation study of the components of our proposed LV-Adapter.

| # | Prompt | V2L | L2V | mIoU (SS) | mIoU (DA) |
|---|---|---|---|---|---|
| 1 | ✗ | ✗ | ✗ | 54.5 | 74.1 |
| 2 | ✗ | ✗ | ✗ | 61.5 | 83.5 |
| 3 | ✓ | ✗ | ✗ | 61.8 | 83.5 |
| 4 | ✓ | ✓ | ✗ | 61.3 | 83.6 |
| 5 | ✓ | ✗ | ✓ | **62.2** | **83.7** |

Table 5: Comparison of different configurations of *adapt* and *finetune* epochs.

| Adapt | Finetune | mIoU (SS) | mIoU (DA) | mAP |
|---|---|---|---|---|
| 12 | 24 | 59.5 | 82.9 | 25.8 |
| 6 | 30 | 61.0 | 83.4 | 26.3 |
| 1 | 35 | 61.5 | 83.5 | 26.4 |
| 6 | 35 | 61.3 | 83.5 | 26.5 |
| 12 | 35 | 61.9 | 83.5 | 26.6 |

adapt epochs have little impact on the results. We hypothesize that fewer adapt stage is enough for the training paradigm since there are few training parameters (only the adapter) during the adapt stage. For a fair comparison with the baseline whose full epochs are 36, we fix the adapt and finetune epochs as 1 and 35 respectively.

## 6 Conclusion

In this paper, we present the first effort on revealing the degradation of state-of-the-art self-supervised models under the multi-task learning scenario in autonomous driving. To reduce the gap between the *pre-training* and *fine-tuning* stage, we propose a simple yet highly efficient *pretrain-adapt-finetune* paradigm, which boosts the performances of different self-supervised pretrained models by a large margin without increasing the overall training overhead. We further utilize the vision-language pre-training model CLIP as a complement to our proposed paradigm and propose LV-Adapter, which incorporates linguistic knowledge into visual features via learning task-specific prompts in a novel fashion. Extensive experiments showcase that (1) the *adapt* stage is the key to mitigate the deficiency of the dominating *pretrain-finetune* paradigm in multi-task learning, and (2) the language priors encoded by CLIP are of general benefits for multiple downstream tasks.

## Acknowledgements

We gratefully acknowledge the support of MindSpore[2], CANN (Computer Architecture for Neural Networks) and Ascend AI Processor used for this research.

## References

[1] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 1877–1901, 2020.

[2] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:9912–9924, 2020.

[3] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning (ICML)*, pages 1597–1607, 2020.

[4] X. Chen, H. Fan, R. Girshick, and K. He. Improved baselines with momentum contrastive learning. *arXiv:2003.04297*, 2020.

[5] B. Cheng, A. Schwing, and A. Kirillov. Per-pixel classification is not all you need for semantic segmentation. *Advances in Neural Information Processing Systems (NeurIPS)*, 34, 2021.

[6] M. Crawshaw. Multi-task learning with deep neural networks: A survey. *arXiv preprint arXiv:2009.09796*, 2020.

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[2] https://www.mindspore.cn/

[8] J. Donahue, P. Krähenbühl, and T. Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016.

[9] J. Donahue and K. Simonyan. Large scale adversarial representation learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019.

[10] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2650–2658, 2015.

[11] C. Fifty, E. Amid, Z. Zhao, T. Yu, R. Anil, and C. Finn. Efficiently identifying task groupings for multi-task learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 34, 2021.

[12] P. Gao, S. Geng, R. Zhang, T. Ma, R. Fang, Y. Zhang, H. Li, and Y. Qiao. Clip-adapter: Better vision-language models with feature adapters. *arXiv preprint arXiv:2110.04544*, 2021.

[13] T. Gao, A. Fisch, and D. Chen. Making pre-trained language models better few-shot learners. In *Association for Computational Linguistics (ACL)*, 2021.

[14] G. Ghiasi, B. Zoph, E. D. Cubuk, Q. V. Le, and T.-Y. Lin. Multi-task self-training for learning general representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8856–8865, 2021.

[15] R. B. Girshick. Fast r-cnn. pages 1440–1448, 2015.

[16] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar, B. Piot, k. kavukcuoglu, R. Munos, and M. Valko. Bootstrap your own latent: a new approach to self-supervised learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:21271–21284, 2020.

[17] X. Gu, T.-Y. Lin, W. Kuo, and Y. Cui. Open-vocabulary object detection via vision and language knowledge distillation. In *International Conference on Learning Representations (ICLR)*, 2022.

[18] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9729–9738, 2020.

[19] K. He, R. B. Girshick, and P. Dollár. Rethinking imagenet pre-training. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4917–4926, 2019.

[20] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[21] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár. Panoptic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9404–9413, 2019.

[22] I. Kokkinos. Ubernet: Training a 'universal' convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6129–6138, 2017.

[23] B. Lester, R. Al-Rfou, and N. Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3045–3059, 2021.

[24] X. L. Li and P. Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[25] V. Likhosherstov, A. Arnab, K. Choromanski, M. Lucic, Y. Tay, A. Weller, and M. Dehghani. Polyvit: Co-training vision transformers on images, videos and audio. *arXiv preprint arXiv:2111.12993*, 2021.

[26] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 936–944, 2017.

[27] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*, 2021.

[28] X. Liu, K. Ji, Y. Fu, Z. Du, Z. Yang, and J. Tang. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*, 2021.

[29] X. Liu, Y. Zheng, Z. Du, M. Ding, Y. Qian, Z. Yang, and J. Tang. Gpt understands, too. *arXiv preprint arXiv:2103.10385*, 2021.

[30] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[31] J. Lu, V. Goswami, M. Rohrbach, D. Parikh, and S. Lee. 12-in-1: Multi-task vision and language representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10437–10446, 2020.

[32] C. Peng, T. Xiao, Z. Li, Y. Jiang, X. Zhang, K. Jia, G. Yu, and J. Sun. Megdet: A large mini-batch object detector. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6181–6189, 2018.

[33] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning (ICML)*, pages 8748–8763, 2021.

[34] M. Raghu, T. Unterthiner, S. Kornblith, C. Zhang, and A. Dosovitskiy. Do vision transformers see like convolutional neural networks? *Advances in Neural Information Processing Systems (NeurIPS)*, 34, 2021.

[35] Y. Rao, W. Zhao, G. Chen, Y. Tang, Z. Zhu, G. Huang, J. Zhou, and J. Lu. Denseclip: Language-guided dense prediction with context-aware prompting. *arXiv preprint arXiv:2112.01518*, 2021.

[36] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 28, 2015.

[37] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[38] T. Schick and H. Schütze. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of NAACL*, 2021.

[39] T. Shin, Y. Razeghi, R. L. Logan IV, E. Wallace, and S. Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*, 2020.

[40] T. S. Standley, A. R. Zamir, D. Chen, L. J. Guibas, J. Malik, and S. Savarese. Which tasks should be learned together in multi-task learning? In *International Conference on Machine Learning (ICML)*, pages 9120–9132, 2020.

[41] F.-K. Sun and C.-I. Lai. Conditioned natural language generation using only unconditioned language model: An exploration. *arXiv preprint arXiv:2011.07347*, 2020.

[42] P. Sun, R. Zhang, Y. Jiang, T. Kong, C. Xu, W. Zhan, M. Tomizuka, L. Li, Z. Yuan, C. Wang, and P. Luo. Sparse r-cnn: End-to-end object detection with learnable proposals. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14454–14463, 2021.

[43] X. Sun, R. Panda, and R. S. Feris. Adashare: Learning what to share for efficient deep multi-task learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:8728–8740, 2020.

[44] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[45] M. Teichmann, M. Weber, M. Zoellner, R. Cipolla, and R. Urtasun. Multinet: Real-time joint semantic reasoning for autonomous driving. In *IEEE Intelligent Vehicles Symposium (IV)*, pages 1013–1020, 2018.

[46] A. van den Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[47] A. Vaswani, N. M. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems (NeurIPS)*, 30, 2017.

[48] X. Wang, R. Zhang, C. Shen, T. Kong, and L. Li. Dense contrastive learning for self-supervised visual pre-training. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3023–3032, 2021.

[49] D. Wu, M. Liao, W. Zhang, and X. Wang. Yolop: You only look once for panoptic driving perception. *arXiv preprint arXiv:2108.11250*, 2021.

[50] T. Xiao, Y. Liu, B. Zhou, Y. Jiang, and J. Sun. Unified perceptual parsing for scene understanding. In *European Conference on Computer Vision (ECCV)*, pages 418–434, 2018.

[51] E. Xie, J. Ding, W. Wang, X. Zhan, H. Xu, Z. Li, and P. Luo. Detco: Unsupervised contrastive learning for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8392–8401, 2021.

[52] Q. Xie, E. H. Hovy, M.-T. Luong, and Q. V. Le. Self-training with noisy student improves imagenet classification. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, 2020.

[53] Z. Xie, Y. Lin, Z. Zhang, Y. Cao, S. Lin, and H. Hu. Propagate yourself: Exploring pixel-level consistency for unsupervised visual representation learning. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16679–16688, 2021.

[54] Z. Yang, Y. Zhang, J. Yu, J. Cai, and J. Luo. End-to-end multi-modal multi-task vehicle control for self-driving cars with visual perceptions. *24th International Conference on Pattern Recognition (ICPR)*, pages 2289–2294, 2018.

[55] Y. Yao, A. Zhang, Z. Zhang, Z. Liu, T.-S. Chua, and M. Sun. Cpt: Colorful prompt tuning for pre-trained vision-language models. *arXiv preprint arXiv:2109.11797*, 2021.

[56] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2636–2645, 2020.

[57] Z. Zhong, D. Friedman, and D. Chen. Factual probing is [mask]: Learning vs. learning to recall. *arXiv preprint arXiv:2104.05240*, 2021.

[58] C. Zhou, C. C. Loy, and B. Dai. Denseclip: Extract free dense labels from clip. *arXiv preprint arXiv:2112.01071*, 2021.

[59] K. Zhou, J. Yang, C. C. Loy, and Z. Liu. Learning to prompt for vision-language models. *arXiv preprint arXiv:2109.01134*, 2021.

[60] X. Zhu, J. Zhu, H. Li, X. Wu, X. Wang, H. Li, X. Wang, and J. Dai. Uni-perceiver: Pre-training unified architecture for generic perception for zero-shot and few-shot tasks. *arXiv preprint arXiv:2112.01522*, 2021.

## Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] See Section Abstract and Introduction.
   (b) Did you describe the limitations of your work? [Yes] See Limitations in Appendix G.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Potential Negative Social Impact in Appendix F.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] Code will be released.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section Experiment Settings and Appendix A.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] Our experiments are quite stable with multiple runs. All results are observed via a fixed seed.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Implementation Details in Section Experiment Setting.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Section Reference.
   (b) Did you mention the license of the assets? [No] The dataset used in the paper is publicly released.
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] The dataset used in the paper is publicly released.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] The dataset used in the paper does not contain personally identifiable information or offensive content.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

# A Experiment Setting

We introduce the details of three settings in our experiments.

**Disjoint-normal setting.** Under the same annotation effort or budget, the quantity of labels decreases with the increase of the complexity of labeling a task. Hence, in this setting, we consider a realistic scenario where each input image is labeled to only one task, namely, the annotations of each task are disjoint, and the number of labeled images for each task is in decreasing order as follows: drivable area segmentation (20k), object detection (10k), semantic segmentation (7k).

**Disjoint-balance setting.** We further consider a more difficult setting with the lowest quantity of annotations that corresponds to the scenarios of scarce annotations. There are 21k images in this setting and each task has 7k labeled images that are not overlapped with other tasks.

**Full-setting.** Full-setting refers to experimenting on all available annotations on $\sim$74k images in BDD100K and can be used to analyze the upper bound of different methods.

# B Revisiting Different Types of Pre-training Methods

**Supervised pre-training** We select the weights pretrained on the ImageNet, which is the most widely adopted pretrained model in the past decade.

**Classification-oriented methods** Currently, the state-of-the-art classification-oriented pre-training methods are mainly based on contrastive learning and online clustering. During the pre-training phase, they produce image-level prediction using global features and minimize the distance between positive pairs while push the representations of negative pairs apart. This eliminates the need for the computation-intensive generation step in previous generative methods [8, 9]. However, they may lack spatial sensitivity for fine-grained tasks since the spatial details are not considered in their formulation. We select MoCo-v1 [18], MoCo-v2 [18], SimCLR [3], SwAV [2], BYOL [16] for comparison.

**Detection-oriented methods** DetCo [51] is specially designed for object detection and enforce contrastive learning between the global image and local patches with multi-level supervision. It achieves well performance on object detection while maintaining competitive classification accuracy.

**Segmentation-oriented methods** Different from the aforementioned ones, this type of method pursues pixel-level self-supervised learning for learning dense feature representations. For example, PixPro [53] propose a pixel-to-propagation consistency task, where two asymmetric pipelines are utilized to obtain positive pixel pairs. We select DenseCL [48] for comparison. We download self-supervised pretrained models from MMSelfSup repository[3].

# C Discussion

We mainly discuss several closely related works on multi-task learning and highlight the differences in between in this section.

**Multi-Task Self-Training**. [14] focuses on the pre-training stage and claims that pre-training on a multi-task pseudo labeled dataset outperforms specialized supervised models and self-supervised models. Some of its findings back up our experiment results in Section 3. In [14], this pre-training paradigm is tailored for downstream **single-task learning** but not multi-task learning. Moreover, MuST conducts cross dataset pretraining on nearly 304M images with an enormous amount of pseudo labels, which incurs a huge computation overhead. In contrast, we show that the performance of a wide range of off-the-shelf pretrained models can be steadily improved without redesigning the resource-intensive pre-training stage or increasing the total training cost. We leave it for future work to explore the influence of this kind of pre-training paradigm on HMPL.

**Multi-Task Models.** We notice that several concurrent works [22, 31, 60, 25] have proposed multi-task models that cover a wide range of tasks. Our models differ notably mainly in two aspects. Firstly, we perform *heterogeneous* multi-task learning where each task has different objectives and evaluation metrics, which increases the optimization difficulty. Secondly, we propose general approaches to

[3] https://github.com/open-mmlab/mmselfsup

Table 6: Comparison with handcrafted prompts under the disjoint-normal setting.

| Method | mAP | AP50 | AP75 | mIoU (SS) | mIoU (DA) |
|---|---|---|---|---|---|
| Prompt engineering | 26.1 | 50.1 | 23.4 | 61.4 | 83.3 |
| Prompt ensembling | 26.3 | 50.4 | 23.5 | 61.7 | 83.3 |
| Learned prompts | **26.4** | **50.5** | **23.7** | **62.2** | **83.7** |

Table 7: Comparisons of self-supervised models under the Disjoint-normal setting with ResNet-50 backbone and convolutional heads.

| Method | mAP | AP50 | AP75 | mIoU (SS) | mIoU (DA) |
|---|---|---|---|---|---|
| Supervised | 23.7 | 46.3 | 20.8 | 58.9 | 83.1 |
| MoCo-v1 | 24.2 | 46.9 | 21.5 | 59.6 | 83.1 |
| MoCo-v2 | 24.5 | 47.3 | 21.9 | 59.4 | 83.4 |
| SimCLR | 24.0 | 46.8 | 21.3 | 58.3 | 83.2 |
| SwAV | 24.0 | 47.5 | 20.7 | 59.7 | 82.8 |
| BYOL | 23.6 | 46.3 | 20.3 | 59.8 | 83.0 |
| DetCo | 24.1 | 46.8 | 21.5 | 59.6 | 83.3 |
| DenseCL | 24.7 | 47.7 | 22.0 | 59.8 | 83.4 |
| PixPro | 24.2 | 46.9 | 21.6 | 60.2 | 83.5 |

improve the overall multi-task performance, while other works propose task-specific or model-specific methods instead.

**Task Grouping.** Task grouping approaches [40, 11] are mainly concerned with determining task groupings to avoid negative transfer between tasks. In this paper, we hypothesize that in the scenario of autonomous driving, the perception tasks are implicitly correlated with each other. From this perspective, task grouping approaches are complementary to ours since once the grouping is decided, the tasks within a group are trained together in a multi-head architecture and our proposed module can be seamlessly integrated.

# D More Ablation Studies

## D.1 Comparison with Handcrafted Prompts

We compare learned prompts with traditionally handcrafted prompts as in Table 6. CLIP [33] designs several handcrafted prompts for zero-shot classification, such as "a photo of [CLASS].". However, these prompts are not suitable for object detection and semantic segmentation. Inspired by [17], we adopt a better prompt "there is a [CLASS] in the scene." for prompt engineering. We follow [33] to ensemble multiple prompt templates, which is also a popular method for zero-shot classification. As shown in Table 6, learned prompts perform better than prompt engineering and prompt ensembling. We conjecture that learned prompts encode more task-related meanings beyond the handcrafted templates.

## D.2 Influence of Head Architecture

To verify the influence of head architectures in multi-task learning, we build a new multi-task model based on Faster R-CNN [36] and Semantic FCN [21], which retains only convolution operators in heads. We report the performance of different pre-training methods in Table 8. As can be seen, their performance becomes much more stable, but is inferior to the transformer-based architecture in Table 2 in paper. The results indicate that the architectural gap is one of the core reasons for the degradation of these models.

To verify the effectiveness of our proposed adapt stage across different architectures, we provide results of the pretrain-adapt-finetune paradigm for popular convolutional heads (Faster R-CNN and Semantic FPN) in Table 8:

Table 8: Comparisons of different training schemes for self-supervised models with convolutional heads.

| Method | Scheme | mIou (SS) | mIoU (DA) | mAP |
|---|---|---|---|---|
| SimCLR | pretrain-finetune | **58.3** | 83.2 | 24.0 |
| | pretrain-adapt-finetune | **58.3** | **83.3** | **24.6** |
| DetCo | pretrain-finetune | 59.6 | 83.3 | 24.1 |
| | pretrain-adapt-finetune | **59.7** | **83.4** | **24.8** |

From the results, we conclude that the pretrain-adapt-finetune paradigm can boost performance on all tasks consistently. However, the improvement is limited, since there is less discrepancy between these CNN-based heads and pretrained models.

### D.3 Influence of Adapter Design

We also carry out additional experiments on different adaptation module designs inspired by [12]. Five more different designs are considered, including 1) pre-adapter: inserting two convolutional layers before the FPN; 2) post-adapter: adding two convolutional layers after the FPN; 3) pre-adapter (learnable scalar): pre-adapter with residual connection by summing the input and output feature weighted by the learnable scalar which is initialized with a small number 0.001; 4) pre-adapter (learnable vector): pre-adapter with residual connection by summing the input and output feature weighted by the learnable vector which is initialized with 0.001. Note that all experiments are trained with the same pseudo labels under the disjoint-normal setting and use the SimCLR pretrained model to initialize the backbone.

Table 9: Comparison of different adapter designs.

| Adapter | mIoU (SS) | mIoU (DA) | mAP |
|---|---|---|---|
| FPN | 60.1 | 83.5 | 25.2 |
| pre-adapter | 60.2 | 83.4 | 25.4 |
| post-adapter | 60.2 | 83.5 | 25.4 |
| pre-adapter (learnable scalar) | 59.9 | 83.6 | 25.5 |
| pre-adapter (learnable vector) | 59.8 | 83.6 | 25.1 |

From the results, we can observe that different adapters achieve comparable results. For simplicity, we choose the simple FPN adapter as the baseline.

### D.4 Language Prior for Single Task

We further conduct the experiments to validate our LV-Adapter equipped the language prior on single-task models under the disjoint-balance setting. Results are shown in Table 10.

Table 10: Comparisons on single-task setting.

| Model | mIoU (SS) | mIoU (DA) | mAP |
|---|---|---|---|
| MaskFormer | 57.1 | - | - |
| + LV-Adaper | **61.3**$^{+4.2\%}$ | - | - |
| MaskFormer | - | 78.1 | - |
| + LV-Adaper | - | **81.6**$^{+3.5\%}$ | - |
| Sparse R-CNN | - | - | 18.6 |
| + LV-Adaper | - | - | **22.4**$^{+3.8\%}$ |

We can see that equipped with language prior, LV-Adapter improves single-task models by a large margin (+4.2% mIoU in semantic segmentation, +3.5% mIoU in drivable area segmentation, and

+3.8% mAP in object detection), indicating that the language knowledge can also serve as a strong prior to enhance the visual representation.

## E Experiments on NuImages Dataset

We further conduct experiments on NuImages dataset[4], which contains full labels on both object detection and semantic segmentation. Results are shown in the following table 11. Note that the multi-task baseline is trained with FPN as the adapter under the pretrain-finetune paradigm.

Table 11: Comparisons of single-task and multi-task models on NuImages dataset.

| Method | mAP | AP50 | AP75 | mIoU |
|---|---|---|---|---|
| Sparse R-CNN | 46.6 | 72.8 | 49.4 | - |
| MaskFormer | - | - | - | 55.8 |
| Multi-task | 47.1 | 73.5 | 49.9 | 53.3 |
| LV-Adapter | **50.3** | **76.8** | **54.3** | **56.0** |

As shown in the table, the multi-task model performs worse than single-task models in semantic segmentation. Our proposed LV-Adapter achieves the best on all tasks, gaining 3.2% mAP and 2.7% mIoU improvement in object detection and semantic segmentation respectively compared with the multi-task baseline. This verifies the effectiveness of our method.

## F Potential Negative Social Impact

Our method has no ethical risk on dataset usage and privacy violation since all benchmarks are public.

## G Limitations

Due to the limited amount and the scarcity of annotations, we only study three tasks in parallel. A more diversified task set with more annotations will further benefit the learning of multi-headed architecture. Besides, since the MuST model [14] is not released and it is impractical for us to conduct resource-intensive multi-task pretraining with limited resources, the influence of multi-task pretraining on multi-task transfer learning is yet to be explored and is out of the scope of this paper.

[4] https://www.nuscenes.org/nuimages