G-NAS: Generalizable Neural Architecture Search for Single Domain Generalization Object Detection

Fan Wu[1], **Jinling Gao**[1], **Lanqing Hong**[2], **Xinbing Wang**[1], **Chenghu Zhou**[1], **Nanyang Ye**[1*]

[1] Shanghai Jiao Tong University, Shanghai, China
[2] Huawei Noah's Ark Lab, Hong Kong, China
wufan55@sjtu.edu.cn, gaojinling@sjtu.edu.cn, honglanqing@huawei.com, xwang8@sjtu.edu.cn, zhouchsjtu@gmail.com, ynylincoln@sjtu.edu.cn

Abstract

In this paper, we focus on a realistic yet challenging task, Single Domain Generalization Object Detection (S-DGOD), where only one source domain's data can be used for training object detectors, but have to generalize multiple distinct target domains. In S-DGOD, both high-capacity fitting and generalization abilities are needed due to the task's complexity. Differentiable Neural Architecture Search (NAS) is known for its high capacity for complex data fitting and we propose to leverage Differentiable NAS to solve S-DGOD. However, it may confront severe over-fitting issues due to the feature imbalance phenomenon, where parameters optimized by gradient descent are biased to learn from the easy-to-learn features, which are usually non-causal and spuriously correlated to ground truth labels, such as the features of background in object detection data. Consequently, this leads to serious performance degradation, especially in generalizing to unseen target domains with huge domain gaps between the source domain and target domains. To address this issue, we propose the Generalizable loss (G-loss), which is an OoD-aware objective, preventing NAS from over-fitting by using gradient descent to optimize parameters not only on a subset of easy-to-learn features but also the remaining predictive features for generalization, and the overall framework is named G-NAS. Experimental results on the S-DGOD urban-scene datasets demonstrate that the proposed G-NAS achieves SOTA performance compared to baseline methods. Codes are available at *https://github.com/wufan-cse/G-NAS*.

Introduction

Object detection is a fundamental task in computer vision (Ren et al. 2015; Tan, Pang, and Le 2020; Ge et al. 2021; Zhang et al. 2021). However, improving the generalization ability of object detection remains a challenging problem, especially for Out-of-Distribution scenarios, where data are sampled from novel unseen distributions. Recently, this has been grounded to a realistic yet challenging task, i.e., Single Domain Generalization Object Detection (S-DGOD) (Wu and Deng 2022), which raised worldwide researchers' attention to the generalization ability of object detection algorithms. The objective of S-DGOD is to improve object detectors' Out-of-Domain (OoD) generalization ability, given a single source domain for training (see Figure 1 for illustration). It requires methods to extract the causal features in the source domain and learn from them for generalization. Compared with the traditional Domain Generalization (DG), S-DGOD provides only one source domain data, making it easy to over-fit as we are unable to learn the features shared by multiple source domains, which usually contain causal information (Arjovsky et al. 2019). Most existing works on S-DGOD (Pan et al. 2018, 2019; Huang et al. 2019; Choi et al. 2021) apply feature normalization to solve the single-domain generalization problem. Other methods, such as feature disentanglement (Wu and Deng 2022) and invariant-based algorithms (Rao et al. 2023), have been proposed. However, none of these existing works discover the high capacity of architectural design in learning complex data distribution. Additionally, there are researches (Ganin et al. 2016; Chen et al. 2018; Saito et al. 2019; Hsu et al. 2020; Chen et al. 2020) on improving object detectors' generalization ability via the Domain Adaption (DA) setting, which learns from a source domain and generalizes to a specific target domain. Compared with DA, S-DGOD targets multiple unseen domains, while DA algorithms solely focus on one target domain and have privileged access to unlabeled target domain data during training, which makes S-DGOD much more challenging than DA.

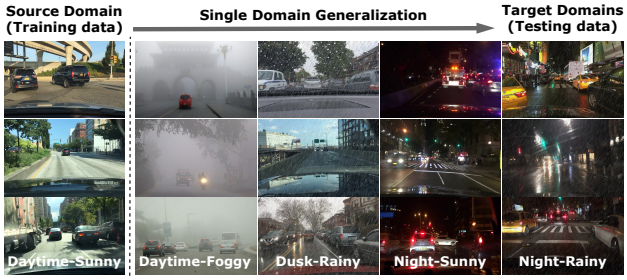

Figure 1: The setting of S-DGOD, which aims to learn from a single source domain and generalize to multiple unseen target domains. It requires extracting the causal features in the source domain for achieving OoD generalization.

In this paper, we propose to leverage the high capacity in fitting complex data of Differentiable NAS to solve

*Nanyang Ye is the corresponding author.

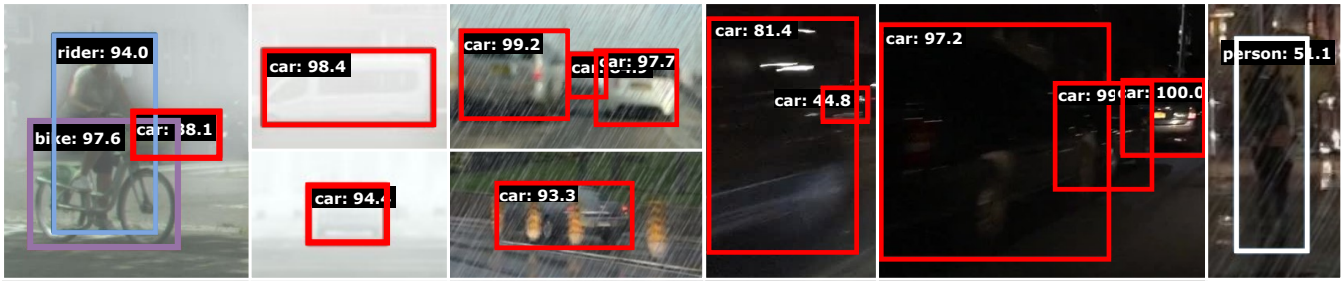

Figure 2: Predictions (category: confidence) of G-NAS on Single Domain Generalization Object Detection tasks. G-NAS can detect objects in extremely-challenging environments. Box color indicates the category. Better zoom-in to view.

the challenging S-DGOD. Here comes the question: Differentiable NAS methods (Liu, Simonyan, and Yang 2018; Yang et al. 2020; Zhong et al. 2020) are known to easily over-fit the training data, how to make them generalizable? DNNs' parameters optimized by gradient descent tend to prioritize learning and making predictions based on easy-to-learn features (Arjovsky et al. 2019; Jacot, Gabriel, and Hongler 2018; Pezeshki et al. 2021). This phenomenon has further implications on Differentiable NAS, which applies gradient descent to learn the optimal architectural parameters, leading to a bias towards optimizing architecture with a few easy-to-learn features. Consequently, Differentiable NAS is significantly affected by the spurious correlations between easy features and labels, and easily over-fits the training data, suffering from sub-optimal OoD performance. To address this issue, we propose Generalizable loss (G-loss), which is OoD-aware, to guide the NAS framework and activate both the network's parameters and architectural parameters to learn from not only the easy features but also the remaining predictive features. Figure 2 shows the superior performance of our proposed G-NAS.

Our main contributions can be summarized as follows:

- To the best of our knowledge, our work is the first attempt to introduce Differentiable NAS for S-DGOD, leveraging the high capacity of NAS methods in fitting complex data features.
- We propose an OoD-aware objective, namely G-loss, to avoid the NAS process from the over-fitting issue, thus, improving OoD generalization performance.
- Extensive experiments demonstrate our proposed G-NAS empirically outperforms previous SOTA baselines on the challenging S-DGOD benchmarks.

Related Works

Single Domain Generalization Object Detection

The majority of works on S-DGOD can be categorized into two main approaches: feature normalization and invariant-based algorithms. IBN-Net (Pan et al. 2018) integrates Instance Normalization (IN) and Batch Normalization (BN) into popular deep neural networks to enhance generalization capacity. Switchable Whitening (SW) (Pan et al. 2019) proposes an approach, which selects appropriate whitening methods to adapt to different tasks. Iterative Normalization (IterNorm) (Huang et al. 2019) employs Newton's iterations to efficiently perform feature normalization. RobustNet (ISW) (Choi et al. 2021) proposes an instance selective whitening loss to disentangle the domain-specific style and domain-invariant feature representations. Cyclic-Disentangled Self-Distillation (CDSD) (Wu and Deng 2022) aims to disentangle domain-invariant representations (DIR) from domain-specific representations and make predictions based on DIR. Style-Hallucinated Dual Consistency Learning (SHADE) (Zhao et al. 2022) proposes two constraints to encourage models to learn from style-diversified samples while keeping them from over-fitting. CLIPGap (Vidit, Engilberge, and Salzmann 2023) leverages the pre-trained knowledge of Vision-Language Models (VLM) to enhance the generalization ability. SRCD (Rao et al. 2023) proposes two novel modules to eliminate the effects of spurious correlation and force model to learn from the semantic relationships. Despite the significant improvements achieved by these works, none of them have explored the potential of NAS algorithms to enhance the capacity of DNNs in fitting complex data distributions.

Neural Architecture Search for Object Detection

Compared with NAS works for the standard image classification tasks, the works of NAS for Object Detection are relatively rare due to their intricacy. Existing works on NAS for Object Detection can be generally divided into three genres according to the searched component in networks, including backbone search (Cai, Zhu, and Han 2018; Chen et al. 2019; Guo et al. 2020; Jiang et al. 2020), Feature Pyramid Network (FPN) search (Ghiasi, Lin, and Le 2019; Liang et al. 2021), and joint detection head and FPN search (Xu et al. 2019; Wang et al. 2020). In the setting of Single-DGOD, these works may end up with sub-optimal OoD generalization performance since they aim to find architectures by minimizing the in-distribution loss. On the contrary, our proposed method is guided by an additional OoD-aware objective, enabling us to identify the architecture with optimal OoD performance.

Methodology

Preliminary on Differentiable NAS

In this paper, we conduct the Differentiable NAS on the prediction head as shown in Figure 3. Conventional Differen-

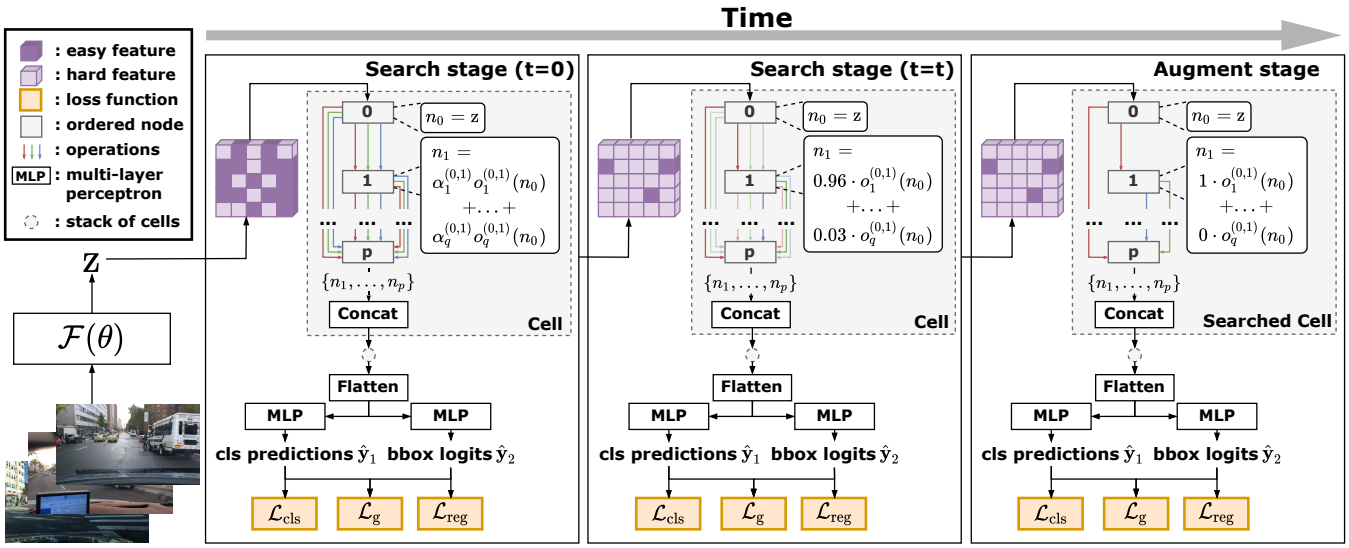

Figure 3: An overview of the proposed G-NAS. At the beginning of the search stage (t=0), the searchable prediction head super-net is randomly initialized, and the feature $\mathbf{z}$ extracted by the detector network $\mathcal{F}(\theta)$ contains both easy and hard features. At the end of the search stage (t=t), the searchable super-net is converged with chosen operation between each node in cells, and the detector network $\mathcal{F}(\theta)$ is forced by $\mathcal{L}_g$ to learning from hard features, eliminating the influence of the spurious correlation between easy features and ground truth labels. At the augment stage, we reconstruct the prediction head with the searched architectural parameters α^* and retrain the whole network.

tiable NAS (Liu, Simonyan, and Yang 2018) aims at utilizing a gradient-based differentiable optimization to search the optimal sub-architecture of the super-net. The super-net $\mathcal{S}(\omega, \alpha)$ is stacked by several cells which are the computation units to be searched during the training process and are formed as a directed acyclic graph (DAG). There are two types of cells, including normal cells and reduction cells, where the difference is whether the feature maps are downsampled or not. A cell is comprised of p ordered nodes $\mathcal{N} = \{n_1, \ldots, n_p\}$ with q candidate operations $\mathcal{O} = \{o_1, \ldots, o_q\}$ between each node. Binary variables $\delta_k^{(i,j)} \in \{0, 1\}$ represents whether candidate operation $o_k^{(i,j)}$ between node n_i and n_j is chosen or not. Thus, we have the following formulations for each node:

$$n_j = \sum_{i=0}^{j-1} \sum_{k=1}^{q} \delta_k^{(i,j)} o_k^{(i,j)}(n_i) = \boldsymbol{\delta}_j^T \mathbf{o}_j, \tag{1}$$

where $\boldsymbol{\delta}_j^T$ and $\mathbf{o}_j$ are vectors formed by $\delta_k^{(i,j)}$ and $o_k(n_i)$ respectively. $\mathcal{F}$ denotes the object detector with network parameters θ, and $\mathbf{z}$ denotes the feature representations extracted by $\mathcal{F}(\theta)$. We use $\mathbf{z}$ to initialize $n_0 = \mathbf{z}$ for the first cell and use the output of the previous cell for the rest of the cells. Practically, DARTS-based (Liu, Simonyan, and Yang 2018) methods convert $\delta_k^{(i,j)}$ into continuous relaxation with a soft-max function to make it differentiable:

$$\alpha_k^{(i,j)} = \exp\left(\delta_k^{(i,j)}\right) / \sum_k \exp\left(\delta_k^{(i,j)}\right), \tag{2}$$

$$n_j = \sum_{i=0}^{j-1} \sum_{k=1}^{q} \alpha_k^{(i,j)} o_k^{(i,j)}(n_i) = \boldsymbol{\alpha}_j^T \mathbf{o}_j, \tag{3}$$

where $\alpha_k^{(i,j)}$ are differentiable and the NAS problem is formulated as the following bi-level optimization problem:

$$\omega^* = \arg\min_{\omega} \mathcal{L}_{\text{train}}\left(\mathcal{S}\left(\mathbf{z}; \omega, \alpha\right)\right), \tag{4}$$

$$\alpha^* = \arg\min_{\alpha} \mathcal{L}_{\text{val}}\left(\mathcal{S}\left(\mathbf{z}; \omega^*, \alpha\right)\right), \tag{5}$$

$$\text{s.t. } \|\alpha_j\|_0 = 1, 1 \le j \le p, \tag{6}$$

where ω is the parameters of the prediction head, $\mathcal{L}_{\text{train}}$ and $\mathcal{L}_{\text{val}}$ are the training loss and validation loss, respectively. During searching process, $\mathcal{L}_{\text{train}}$ and $\mathcal{L}_{\text{val}}$ are optimized alternately (Liu, Simonyan, and Yang 2018). In this paper, we use $\mathcal{L}_{\text{train}}$ to optimize α as the in-domain (i.d.) validation set is not suitable for S-DGOD as we aim to improve OoD generalization ability instead of selecting models with optimal i.d. performance. When we get α^*, the index of the maximum value in $\alpha^{*(i,j)} \in \mathbb{R}^q, 1 \le i < j \le p$ is the chosen operation, then we reconstruct the prediction head and retrain the whole network. For the design of the search space, please refer to Appendix.

Generalizable Objective

As revealed by Arjovsky et al. (2019), Jacot, Gabriel, and Hongler (2018), and Pezeshki et al. (2021), DNNs' parameters optimized by gradient descent exhibit an inclination to learn and make predictions based on the easy-to-learn features. These easy-to-learn features are typically non-causal, such as color blocks. For example, as the GradCam maps (Selvaraju et al. 2017) shown in Figure 4, DNNs could be misled by large salient white blocks and generate false car detection with high confidence. This origins from the spurious correlation between easy-to-learn features (color blocks)

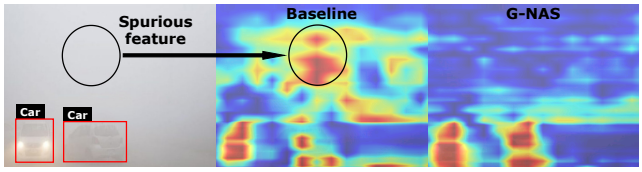

Figure 4: GradCam visualizations on the Daytime-Foggy test set. The results show that the background features significantly mislead the predictions of baseline DNNs, while G-NAS learns the object-related features to make predictions.

and ground truth labels (car annotations) in the daytime training set. Consequently, the remaining features that might have causal correlations with the ground truth labels are disregarded. Especially in the setting of S-DGOD, the OoD generalization is hardly achieved as the spurious correlation between easy features and labels learned in the source domain may not exist in the target domains, where the domain gap between source and target domains is huge. This further hinders differentiable NAS and leads to the inclination that only a subset of architectural choices are activated ignoring the remaining architectures, which may possess more significant generalization capabilities in object detection, and resulting in over-fitting. To address this issue, we propose an OoD-aware objective, Gerneralizable loss (G-loss). The goal of G-loss is to discourage using few dominant network's parameters and architectural candidates to make predictions during training, forcing the DNNs and Differentiable NAS to use more abundant information in representation learning. G-loss takes into account the regression branch of the detection network and architectural parameters to regularize the training process. Our calculation in Theorem 1 leads to the following compact form of G-loss:

$$\mathcal{L}_{\text{g}}(\theta, \omega, \alpha) = \frac{1}{2}\|\hat{\mathbf{y}}_1\|^2 - \frac{1}{2}\|\hat{\mathbf{y}}_2\|^2, \tag{7}$$

where $\hat{\mathbf{y}}_1$ and $\hat{\mathbf{y}}_2$ are the outputs of the classification head and the regression head, respectively. We now discuss how G-loss operates to promote balance training. We assume the width of DNNs goes infinite, in the regime of Neural Tangent Kernel (NTK) theory (Jacot, Gabriel, and Hongler 2018), we have the following proposition:

Proposition 1. ***(NTRF approximation of DNNs.)*** *When the width of neural networks goes infinite, the output of over-parameterized neural networks can be approximated as a linear function:*

$$\hat{\mathbf{y}}_1 = \psi \cdot \Theta \cdot w_1, \tag{8}$$
$$\hat{\mathbf{y}}_2 = \psi \cdot \Theta \cdot w_2, \tag{9}$$

where $\psi \in \mathbb{R}^{n\times m}$ is the Neural Tangent Random Feature (NTRF) matrix (Cao and Gu 2019) of n training data, $\Theta \in \mathbb{R}^m$ denotes the concatenation of all vectorized trainable parameters with size m, $w_1 \in \mathbb{R}$ and $w_2 \in \mathbb{R}$ project features into classification output $\hat{\mathbf{y}}_1 \in \mathbb{R}^n$ and regression output $\hat{\mathbf{y}}_2 \in \mathbb{R}^n$, respectively.

Based on this proposition, we have the following theorem:

Theorem 1. *Assume the width of the neural network goes infinite. We consider G-loss regularized regression loss $\mathcal{L}_{\text{reg}}$ and classification loss $\mathcal{L}_{\text{cls}}$:*

$$\mathcal{L}(\Theta) = \mathcal{L}_{\text{cls}} + \mathcal{L}_{\text{reg}} + \mathcal{L}_{\text{g}}, \tag{10}$$

where we apply cross-entropy function for $\mathcal{L}_{\text{cls}}$ and smooth L1 function for $\mathcal{L}_{\text{reg}}$. The optimization problem:

$$\begin{aligned}\min_{\Theta} \mathcal{L}(\Theta) = \mathbf{1} \cdot \log\left[1 + \exp\left(-\boldsymbol{Y}_1 \hat{\mathbf{y}}_1\right)\right] \\ + \frac{1}{2}\left(\hat{\mathbf{y}}_2 - \mathbf{y}_2\right)^T \left(\hat{\mathbf{y}}_2 - \mathbf{y}_2\right) \\ + \frac{1}{2}\|\hat{\mathbf{y}}_1\|^2 - \frac{1}{2}\|\hat{\mathbf{y}}_2\|^2,\end{aligned} \tag{11}$$

where the last two terms are $\mathcal{L}_{\text{g}}$, $\boldsymbol{Y}_1 = \text{diag}(\mathbf{y}_1) \in \mathbb{R}^{n\times n}$ is the diagonal matrix of ground truth classification labels $\mathbf{y}_1 \in \mathbb{R}^n$, and $\mathbf{y}_2 \in \mathbb{R}^n$ is the ground truth regression labels [1]. $\mathbf{1}$ denotes the all-ones vector with size n. We only consider the interval $[-1, 1]$ for smooth L1 function [2]. The above optimization problem can be transferred to the following maximization problem on the dual variable:

$$\min_{\Theta} \mathcal{L}(\Theta) = \max_{\Phi} \mathcal{H}(\Phi), \tag{12}$$

$$\begin{aligned}\mathcal{H}(\Phi) = -\mathbf{1} \cdot [\Phi \log \Phi + (1-\Phi)\log(1-\Phi)] \\ - \Phi^T \boldsymbol{Y}_1 \psi \Delta w_1 + \frac{1}{2}\left(\psi\Delta w_2 - \mathbf{y}_2\right)^T \left(\psi\Delta w_2 - \mathbf{y}_2\right) \\ + \frac{1}{2}\|\psi\Delta w_1\|^2 - \frac{1}{2}\|\psi\Delta w_2\|^2,\end{aligned} \tag{13}$$

$$\Delta = \psi^{-1}\left(\frac{1}{w_1}\boldsymbol{Y}_1^T\Phi + \frac{w_2}{w_1^2}\mathbf{y}_2\right), \tag{14}$$

where $\Phi \in (0,1)^n$ is a variational parameter defined for each training example. Together with Proposition 1, the gradient descent for Φ is calculated as follows:

$$\frac{\partial \mathcal{H}(\Phi)}{\partial \Phi} = \log\frac{1-\Phi}{\Phi} - \boldsymbol{Y}_1\boldsymbol{Y}_1^T\Phi - \frac{w_2}{w_1}\boldsymbol{Y}_1\mathbf{y}_2. \tag{15}$$

Note that $\boldsymbol{Y}_1\boldsymbol{Y}_1^T$ is a diagonal matrix, and the overall calculation avoids interference between different elements of Φ in Equation (15) during gradient descent, making training data independent between each other, thus, avoiding the appearance of dominant features and encourages activating more features for predicting. As a result, the whole network is optimized without bias by easy features, learning from not only easy features but also the remaining features with the additional regularization term defined in Equation (7). Proof of Theorem 1 can be found in Appendix.

Algorithm Framework

Our proposed G-NAS is outlined in Algorithm 1 and the structure is depicted in Figure 3. The whole algorithm is

[1] To simplify, we consider each image contains one bounding box and one can easily expand it to multiple bounding boxes following the proof of this theorem.

[2] We can easily use the normalization to constrain the input of smooth L1 function.

Algorithm 1: G-NAS: Generalizable Neural Architecture Search for Single Domain Generalization Object Detection

Require: Training set $\mathcal{D}_{\text{train}}$, generalizable loss weight λ_{g}, learning rate β.

Ensure: A neural architecture with optimized parameters θ^*, ω^* and α^*.

```
# Search stage
Initialize the detector network F(θ) ;
Initialize the searchable prediction head S(ω, α) ;
for each (x, y) ∈ D_train do
    Calculate L_g(θ, ω, α) according to Equation (7) ;
    Calculate L_train according to Equation (16);
    Update θ, ω, α through SGD(L_train, β) algorithm ;
Save the searched architecture α* ;
# Augment stage
Initialize detector network F(θ) ;
Reconstruct the searched prediction head S(ω, α*) ;
for each (x, y) ∈ D_train do
    Calculate L_g(θ, ω, α*) according to Equation (7) ;
    Calculate L_train according to Equation (16);
    Update θ, ω through SGD(L_train, β) algorithm ;
Save the optimized parameters θ*, ω* ;
```

built upon Faster R-CNN (Ren et al. 2015) and contains two stages: the search stage and the augment stage. Firstly, in the search stage, a super-net prediction head $\mathcal{S}(\omega, \alpha)$ is constructed according to Equation (2). We repeat to update the trainable parameters using the Stochastic Gradient Descent algorithm. In the augment stage, we apply the searched architectural parameters α^* to reconstruct the prediction head $\mathcal{S}(\omega, \alpha^*)$, where we only construct the chosen operation. The visualization of the searched architectures can be found in Appendix. For both stages, the training loss $\mathcal{L}_{\text{train}}$ is calculated as followed:

$$\mathcal{L}_{\text{train}} = \mathcal{L}_{\text{det}} + \mathcal{L}_{\text{cls}} + \mathcal{L}_{\text{reg}} + \lambda_g \cdot \mathcal{L}_g, \tag{16}$$

where $\mathcal{L}_{\text{det}}$ is the detector loss (Ren et al. 2015) for the region proposal network (RPN), $\mathcal{L}_{\text{cls}}$ is the cross-entropy loss used for classification, $\mathcal{L}_{\text{reg}}$ is the smooth L1 loss used for bounding box regression, and λ_{g} is a hyper-parameter to determine the weight of $\mathcal{L}_{\text{g}}$.

Experiments

Experimental Setup

Datasets. To evaluate different methods' single-domain generalization ability, we follow the setting proposed by Wu and Deng (2022). The dataset contains five urban-scene domains with distinct weather conditions, including Daytime-Sunny, Daytime-Foggy, Dusk-Rainy, Night-Sunny, and Night-Rainy. The Daytime-Sunny is the source training domain and the other four domains are only used for testing. More details about the construction of these domains can be found in Appendix.

Baselines. We choose eight classic and SOTA algorithms from the Single-DGOD benchmarks (Wu and Deng 2022) for comparison, including IBN-Net (Pan et al. 2018), Switchable Whitening (SW) (Pan et al. 2019), Iterative Normalization (IterNorm) (Huang et al. 2019), RobustNet (ISW) (Choi et al. 2021), Cyclic-Disentangled Self-Distillation (CDSD) (Wu and Deng 2022), Style-Hallucinated Dual Consistency Learning (SHADE) (Zhao et al. 2022), CLIPGap (Vidit, Engilberge, and Salzmann 2023), and SRCD (Rao et al. 2023). For CLIPGap, we use the version initialized with the ImageNet pre-trained weights for fair comparisons as all other baseline methods solely apply ImageNet pre-training.

Evaluation Metric. For all quantitative experiments, we follow Wu and Deng (2022) to evaluate methods' performance using Mean Average Precision (mAP) and report the AP of each class, which is known as PASCAL VOC evaluation metric (Everingham et al. 2010).

Implemetation Details. We apply the Faster R-CNN detector (Ren et al. 2015) with ResNet-101 backbone (He et al. 2016) as the base model for all algorithms to perform object detection. All algorithms are solely initialized by the ImageNet pre-trained weights. We replace the prediction head with our proposed NAS framework, which contains a stem convolution layer, a searchable normal cell, and a searchable reduction cell. The whole training process consists of two sequentially executing stages, the search stage, and the augment stage. We first construct a searchable super-net in the search stage to perform the architecture search within the super-net and save the architecture searched in the last epoch. In the second stage, we reconstruct the prediction head with the architectural parameters obtained in the first stage and perform end-to-end training. We train all models until full convergence for 12 epochs. We set the λ_{g} to 1.0. All parameters in our NAS framework are randomly initialized. We apply an SGD optimizer with the learning rate set to 0.02 and we set the batch size to 4 per GPU. All experiments are conducted on a computer with 8 GPUs.

Comparison with the State of the Art

Overall Single-DGOD Results. Table 1 shows the mAP results on all domains, including Daytime-Sunny, Daytime-Foggy, Dusk-Rainy, Night-Sunny, and Night-Rainy, in which Daytime-Sunny is for training. The average mAPs on test domains are also reported to measure generalization abilities. As shown in Table 1, our proposed G-NAS significantly improves the average generalization performance to 33.5% compared with the SOTA algorithm, i.e., SRCD, which achieves 29.6%. Notably, G-NAS simultaneously achieves SOTA performance on all target domains, where the domain gaps between the source domain (Daytime-Sunny) and each of these target domains are large. This suggests G-NAS is more generalizable and robust against domain shifts compared with baselines.

Daytime-Sunny to Daytime-Foggy. In the Daytime-Foggy scenario, objects are covered by fog, posing extremely challenging test environments. Table 2 lists per-class results on Daytime-Foggy. Notably, G-NAS brings the AP of person class up to 38.6%, surpassing baselines by 5.2%.

Method	Daytime-Sunny	Daytime-Foggy	Dusk-Rainy	Night-Sunny	Night-Rainy	Average
Faster R-CNN (2015)	51.1	31.9	26.6	33.5	14.5	26.6
IBN-Net (2018)	49.7	29.6	26.1	32.1	14.3	25.5
SW (2019)	50.6	30.8	26.3	33.4	13.7	26.1
IterNorm (2019)	43.9	28.4	22.8	29.6	12.6	23.4
ISW (2021)	51.3	31.8	25.9	33.2	14.1	26.3
CDSD (2022)	56.1	33.5	28.2	36.6	16.6	28.7
SHADE (2022)	-	33.4	29.5	33.9	16.8	28.4
CLIPGap (2023)	48.1	32.0	26.0	34.4	12.4	26.2
SRCD (2023)	-	35.9	28.8	36.7	17.0	29.6
G-NAS (Ours)	**58.4**	**36.4**	**35.1**	**45.0**	**17.4**	**33.5**

Table 1: Single domain generalization object detection results. All algorithms are trained on the Daytime-Sunny domain and tested on the other four domains. Average results are calculated using four out-of-domain results to compare the Out-of-Domain generalization ability. All baseline results are mainly taken from SHADE (Zhao et al. 2022), CLIPGap (Vidit, Engilberge, and Salzmann 2023), and SRCD (Rao et al. 2023). "-" denotes the results without reporting in the original paper. The numbers in bold and underlined denote the highest and the second performance, respectively. The results demonstrate that our approach is robust against domain shifts and achieves the SOTA S-DGOD performance.

	Daytime-Foggy							Dusk-Rainy						
Method	**Bus**	**Bike**	**Car**	**Motor**	**Person**	**Rider**	**Truck**	**Bus**	**Bike**	**Car**	**Motor**	**Person**	**Rider**	**Truck**
FR	30.7	26.7	49.7	26.2	30.9	35.5	23.2	36.8	15.8	50.1	12.8	18.9	12.4	39.5
IBN-Net	29.9	26.1	44.5	24.4	26.2	33.5	22.4	37.0	14.8	50.3	11.4	17.3	13.3	38.4
SW	30.6	26.2	44.6	25.1	30.7	34.6	23.6	35.2	16.7	50.1	10.4	20.1	13.0	38.8
IterNorm	29.7	21.8	42.4	24.4	26.0	33.3	21.6	32.9	14.1	38.9	11.0	15.5	11.6	35.7
ISW	29.5	26.4	49.2	27.9	30.7	34.8	24.0	34.7	16.0	50.0	11.1	17.8	12.6	38.8
CDSD	32.9	28.0	48.8	29.8	32.5	38.2	24.1	37.1	19.6	50.9	13.4	19.7	16.3	40.7
SRCD	**36.4**	30.1	52.4	31.3	33.4	**40.1**	**27.7**	39.5	21.4	50.6	11.9	20.1	17.6	40.5
G-NAS	32.4	**31.2**	**57.7**	**31.9**	**38.6**	38.5	24.5	**44.6**	**22.3**	**66.4**	**14.7**	**32.1**	**19.6**	**45.8**

Table 2: Per-class results on Daytime-Foggy and Dusk-Rainy. FR denotes Faster R-CNN.

Compared to other objects, the person class object is generally smaller and much more difficult to detect under adverse weather conditions, while in autonomous driving, it is crucial to accurately detect pedestrians on roads. As shown in Figure 4, G-NAS solves this problem by avoiding fitting spurious correlations that exist in the training data. This result demonstrates that G-NAS is effective in detecting difficult objects, serving for life-critical applications, such as autonomous driving.

Daytime-Sunny to Dusk-Rainy. Dusk-Rainy significantly differs from the source domain, not only in the change of the weather but also in the time. In rainy scenes, vehicles' light will be reflected on objects' surface by the water on the ground, causing changes in objects' appearance and making them harder to identify. As shown in Table 2, our method improves APs of most vehicle classes and achieves SOTA performance.

Daytime-Sunny to Night-Sunny. The night scenarios have been challenging in various research as the lighting condition is too bad to clearly identify completed objects. Specifically, if DNN learns to predict based on spurious correlations, such as the color of cars, its performance might confront severe degeneration as the color is not salient in the night scenario compared with daytime. Table 1 and Table 3 shows that our proposed G-NAS significantly surpasses SOTA baselines by 8.3% mAP. These results demonstrate the feature representations learned by G-NAS are more generalizable, overcoming the influence of spurious correlations.

Daytime-Sunny to Night-Rainy. Night-Rainy is a challenging scenario as the low-light condition is synergized with the rainy weather. Table 3 shows our method achieves the best performance in pedestrian detection and our method achieves 17.4% mAP in Table 1, outperforming SOTA baselines by 0.4%. These results demonstrate that G-NAS is robust even in extremely bad conditions.

Ablation Study

NAS. For the experiments without NAS, We randomly initialize the α and fix the architecture during the whole training process. Table 4 shows that our NAS strategy brings the average performance up to 33.5% mAP with G-loss and 28.2% mAP without G-loss, surpassing G-NAS without NAS by 2.4% mAP and 1.2%, respectively. These results show that our NAS strategy plays a crucial role in improving the generalization performance, demonstrating the

Method	Night-Sunny							Night-Rainy						
	Bus	Bike	Car	Motor	Person	Rider	Truck	Bus	Bike	Car	Motor	Person	Rider	Truck
FR	37.7	30.6	49.5	15.4	31.5	28.6	40.8	22.6	11.5	27.7	0.4	10.0	10.5	19.0
IBN-Net	37.8	27.3	49.6	15.1	29.2	27.1	38.9	24.6	10.0	28.4	0.9	8.3	9.8	18.1
SW	38.7	29.2	49.8	16.6	31.5	28.0	40.2	22.3	7.8	27.6	0.2	10.3	10.0	17.7
IterNorm	38.5	23.5	38.9	15.8	26.6	25.9	38.1	21.4	6.7	22.0	0.9	9.1	10.6	17.6
ISW	38.5	28.5	49.6	15.4	31.9	27.5	41.3	22.5	11.4	26.9	0.4	9.9	9.8	17.5
CDSD	40.6	35.1	50.7	19.7	34.7	32.1	43.4	24.4	11.6	29.5	**9.8**	10.5	11.4	19.2
SRCD	43.1	32.5	52.3	20.1	34.8	31.5	42.9	26.5	**12.9**	32.4	0.8	10.2	**12.5**	**24.0**
G-NAS	**46.9**	**40.5**	**67.5**	**26.5**	**50.7**	**35.4**	**47.8**	**28.6**	9.8	**38.4**	0.1	**13.8**	9.8	21.4

Table 3: Per-class results on Night-Sunny and Night-Rainy.

Method	NAS	G-loss	D-F	D-R	N-S	N-R	Avg.
G-NAS	✘	✘	32.3	27.0	33.9	14.8	27.0
G-NAS	✘	✔	34.6	31.9	40.7	17.0	31.1
G-NAS	✔	✘	33.6	28.0	35.2	16.1	28.2
G-NAS	✔	✔	**36.4**	**35.1**	**45.0**	**17.4**	**33.5**

Table 4: Ablation study on G-NAS. D, F, R, N, and S represent Daytime, Foggy, Rainy, Night, and Sunny, respectively. Avg. denotes the average performance on the four domains.

architectural design has a significant influence on networks' generalization ability.

G-loss. For the ablation study on G-loss, we simply set the λ_g to 0 to eliminate the influence of G-loss. As shown in Table 4, G-loss brings the average performance up to 33.5% mAP with NAS and 31.1% without NAS, surpassing G-NAS without G-loss by 2.5% mAP and 4.1%, respectively. These results reveal that G-loss is efficient for guiding the NAS framework to find the architecture with optimal generalization performance by activating the whole network to learn from causal features. We also conduct an ablation study on the hyper-parameter λ_g in Appendix, and the results show that G-NAS achieves the best OoD performance when λ_g is set to 1, which aligns well with previous theoretical analysis in Theorem 1.

Visualization

Figure 5 shows that our NAS framework trained with G-loss extracts similar feature representations on source and target domains, which indicates the learned features are generalizable and causal as they are stable against domain shifts. On the contrary, the NAS framework trained without G-loss over-fits to the Daytime-Sunny domain, generalizing to the Daytime-Foggy domain but inconsistently performing in the remaining target domains.

Conclusion

In this work, we primarily focus on the challenging S-DGOD scenario, which holds substantial real-world significance. S-DGOD involves training object detectors on a single source domain and enabling them to generalize to numerous unseen target domains. To address this challenge, we harness the potent capacity of NAS techniques to model intricate data distributions. Additionally, We introduce an OoD-aware objective, termed G-loss, to augment the NAS framework in learning crucial information. The experimental results highlight that our proposed approach, G-NAS, surpasses state-of-the-art methods across four distinct weather conditions in S-DGOD benchmarks. Furthermore, our ablation study underscores the indispensability of each proposed module for achieving robust generalization performance. To the best of our knowledge, this study marks the pioneering attempt to tackle NAS in S-DGOD, yielding state-of-the-art performance concurrently.

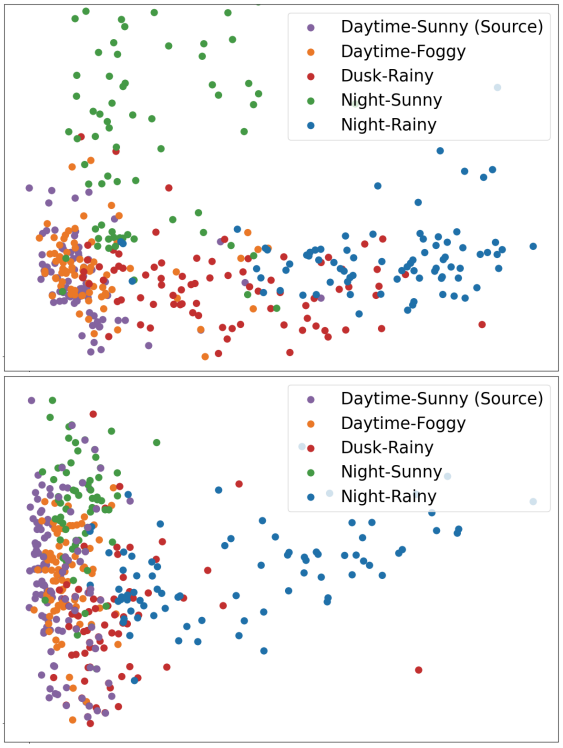

Figure 5: PCA projections of the representations on different domains. The feature representations learned with G-loss (bottom) have more similar patterns across different domains than without G-loss (top).

Acknowledgments

Nanyang Ye was supported in part by National Natural Science Foundation of China under Grant No.62106139, 61960206002, 62272301, 62020106005, 62061146002, 62032020, in part by Shanghai Artificial Intelligence Laboratory and the National Key R&D Program of China under Grant No.2022ZD0160100.

References

Arjovsky, M.; Bottou, L.; Gulrajani, I.; and Lopez-Paz, D. 2019. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*.

Bai, H.; Zhou, F.; Hong, L.; Ye, N.; Chan, S.-H. G.; and Li, Z. 2021. Nas-ood: Neural architecture search for out-of-distribution generalization. In *ICCV*.

Cai, H.; Zhu, L.; and Han, S. 2018. Proxylessnas: Direct neural architecture search on target task and hardware. *arXiv preprint arXiv:1812.00332*.

Cao, Y.; and Gu, Q. 2019. Generalization bounds of stochastic gradient descent for wide and deep neural networks. In *NeurIPS*.

Chen, C.; Zheng, Z.; Ding, X.; Huang, Y.; and Dou, Q. 2020. Harmonizing transferability and discriminability for adapting object detectors. In *CVPR*.

Chen, Y.; Li, W.; Sakaridis, C.; Dai, D.; and Van Gool, L. 2018. Domain adaptive faster r-cnn for object detection in the wild. In *CVPR*.

Chen, Y.; Yang, T.; Zhang, X.; Meng, G.; Xiao, X.; and Sun, J. 2019. Detnas: Backbone search for object detection. In *NeurIPS*.

Choi, S.; Jung, S.; Yun, H.; Kim, J. T.; Kim, S.; and Choo, J. 2021. Robustnet: Improving domain generalization in urban-scene segmentation via instance selective whitening. In *CVPR*.

Cordts, M.; Omran, M.; Ramos, S.; Rehfeld, T.; Enzweiler, M.; Benenson, R.; Franke, U.; Roth, S.; and Schiele, B. 2016. The cityscapes dataset for semantic urban scene understanding. In *CVPR*.

Everingham, M.; Van Gool, L.; Williams, C. K.; Winn, J.; and Zisserman, A. 2010. The pascal visual object classes (voc) challenge. *IJCV*.

Ganin, Y.; Ustinova, E.; Ajakan, H.; Germain, P.; Larochelle, H.; Laviolette, F.; Marchand, M.; and Lempitsky, V. 2016. Domain-adversarial training of neural networks. *JMLR*.

Ge, Z.; Liu, S.; Wang, F.; Li, Z.; and Sun, J. 2021. Yolox: Exceeding yolo series in 2021. *arXiv preprint arXiv:2107.08430*.

Ghiasi, G.; Lin, T.-Y.; and Le, Q. V. 2019. Nas-fpn: Learning scalable feature pyramid architecture for object detection. In *CVPR*.

Guo, Z.; Zhang, X.; Mu, H.; Heng, W.; Liu, Z.; Wei, Y.; and Sun, J. 2020. Single path one-shot neural architecture search with uniform sampling. In *ECCV*.

Hassaballah, M.; Kenk, M. A.; Muhammad, K.; and Minaee, S. 2020. Vehicle detection and tracking in adverse weather using a deep learning framework. *IEEE transactions on intelligent transportation systems*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*.

Hsu, H.-K.; Yao, C.-H.; Tsai, Y.-H.; Hung, W.-C.; Tseng, H.-Y.; Singh, M.; and Yang, M.-H. 2020. Progressive domain adaptation for object detection. In *WACV*.

Huang, L.; Zhou, Y.; Zhu, F.; Liu, L.; and Shao, L. 2019. Iterative normalization: Beyond standardization towards efficient whitening. In *CVPR*.

Huang, Z.; Wang, H.; Xing, E. P.; and Huang, D. 2020. Self-challenging improves cross-domain generalization. In *ECCV*.

Jaakkola, T. S.; and Haussler, D. 1999. Probabilistic kernel regression models. In *Seventh International Workshop on Artificial Intelligence and Statistics*. PMLR.

Jacot, A.; Gabriel, F.; and Hongler, C. 2018. Neural tangent kernel: Convergence and generalization in neural networks. In *NeurIPS*.

Jiang, C.; Xu, H.; Zhang, W.; Liang, X.; and Li, Z. 2020. SP-NAS: Serial-to-parallel backbone search for object detection. In *CVPR*.

Kim, T.; Jeong, M.; Kim, S.; Choi, S.; and Kim, C. 2019. Diversify and match: A domain adaptive representation learning paradigm for object detection. In *CVPR*.

Lee, J.; Xiao, L.; Schoenholz, S.; Bahri, Y.; Novak, R.; Sohl-Dickstein, J.; and Pennington, J. 2019. Wide neural networks of any depth evolve as linear models under gradient descent. In *NeurIPS*.

Li, W.; Li, F.; Luo, Y.; Wang, P.; et al. 2020. Deep domain adaptive object detection: A survey. In *SSCI*.

Liang, T.; Wang, Y.; Tang, Z.; Hu, G.; and Ling, H. 2021. Opanas: One-shot path aggregation network architecture search for object detection. In *CVPR*.

Liu, H.; Simonyan, K.; and Yang, Y. 2018. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*.

Melis, G.; Dyer, C.; and Blunsom, P. 2017. On the state of the art of evaluation in neural language models. *arXiv preprint arXiv:1707.05589*.

Oza, P.; Sindagi, V. A.; Sharmini, V. V.; and Patel, V. M. 2023. Unsupervised domain adaptation of object detectors: A survey. *PAMI*.

Pan, X.; Luo, P.; Shi, J.; and Tang, X. 2018. Two at once: Enhancing learning and generalization capacities via ibn-net. In *ECCV*.

Pan, X.; Zhan, X.; Shi, J.; Tang, X.; and Luo, P. 2019. Switchable whitening for deep representation learning. In *ICCV*.

Pezeshki, M.; Kaba, O.; Bengio, Y.; Courville, A. C.; Precup, D.; and Lajoie, G. 2021. Gradient starvation: A learning proclivity in neural networks. In *NeurIPS*.

Rao, Z.; Guo, J.; Tang, L.; Huang, Y.; Ding, X.; and Guo, S. 2023. SRCD: Semantic Reasoning with Compound Domains for Single-Domain Generalized Object Detection. *arXiv preprint arXiv:2307.01750*.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*.

Review, N. L. 2021. The Dangers of Driverless Cars. https://www.natlawreview.com/article/dangers-driverless-cars. May 5, 2021.

Rodriguez, A. L.; and Mikolajczyk, K. 2019. Domain adaptation for object detection via style consistency. *arXiv preprint arXiv:1911.10033*.

Saito, K.; Ushiku, Y.; Harada, T.; and Saenko, K. 2019. Strong-weak distribution alignment for adaptive object detection. In *CVPR*.

Sakaridis, C.; Dai, D.; and Van Gool, L. 2018. Semantic foggy scene understanding with synthetic data. *IJCV*.

Selvaraju, R. R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; and Batra, D. 2017. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*.

Tan, M.; Pang, R.; and Le, Q. V. 2020. Efficientdet: Scalable and efficient object detection. In *CVPR*.

Vidit, V.; Engilberge, M.; and Salzmann, M. 2023. CLIP the Gap: A Single Domain Generalization Approach for Object Detection. In *CVPR*.

Wang, M.; and Deng, W. 2018. Deep visual domain adaptation: A survey. *Neurocomputing*.

Wang, N.; Gao, Y.; Chen, H.; Wang, P.; Tian, Z.; Shen, C.; and Zhang, Y. 2020. NAS-FCOS: Fast neural architecture search for object detection. In *CVPR*.

Wu, A.; and Deng, C. 2022. Single-domain generalized object detection in urban scene via cyclic-disentangled self-distillation. In *CVPR*.

Wu, A.; Liu, R.; Han, Y.; Zhu, L.; and Yang, Y. 2021. Vector-decomposed disentanglement for domain-invariant object detection. In *ICCV*.

Xu, H.; Yao, L.; Zhang, W.; Liang, X.; and Li, Z. 2019. Auto-fpn: Automatic network architecture adaptation for object detection beyond classification. In *ICCV*.

Yang, Y.; Li, H.; You, S.; Wang, F.; Qian, C.; and Lin, Z. 2020. Ista-nas: Efficient and consistent neural architecture search by sparse coding. In *NeurIPS*.

Yu, F.; Chen, H.; Wang, X.; Xian, W.; Chen, Y.; Liu, F.; Madhavan, V.; and Darrell, T. 2020. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *CVPR*.

Zhang, H.; Wang, Y.; Dayoub, F.; and Sunderhauf, N. 2021. Varifocalnet: An iou-aware dense object detector. In *CVPR*.

Zhao, Y.; Zhong, Z.; Zhao, N.; Sebe, N.; and Lee, G. H. 2022. Style-Hallucinated Dual Consistency Learning: A Unified Framework for Visual Domain Generalization. *arXiv preprint arXiv:2212.09068*.

Zhong, Y.; Deng, Z.; Guo, S.; Scott, M. R.; and Huang, W. 2020. Representation sharing for fast object detector search and beyond. In *ECCV*.

Appendix

Proof of Theorem 1

We first restate the proposition:

Proposition 1. ***(NTRF approximation of DNNs.)*** *When the width of neural networks goes infinite, the output of over-parameterized neural networks can be approximated as a linear function:*

$$\hat{\mathbf{y}}_1 = \psi \cdot \Theta \cdot w_1, \tag{8}$$
$$\hat{\mathbf{y}}_2 = \psi \cdot \Theta \cdot w_2, \tag{9}$$

where $\psi \in \mathbb{R}^{n\times m}$ is the Neural Tangent Random Feature (NTRF) matrix (Cao and Gu 2019) of n training data, $\Theta \in \mathbb{R}^m$ denotes the concatenation of all vectorized trainable parameters with size m, $w_1 \in \mathbb{R}$ and $w_2 \in \mathbb{R}$ project features into classification output $\hat{\mathbf{y}}_1 \in \mathbb{R}^n$ and regression output $\hat{\mathbf{y}}_2 \in \mathbb{R}^n$, respectively.

We then restate the theorem:

Theorem 1. *Assume the width of the neural network goes infinite. We consider G-loss regularized regression loss $\mathcal{L}_{\text{reg}}$ and classification loss $\mathcal{L}_{\text{cls}}$:*

$$\mathcal{L}(\Theta) = \mathcal{L}_{\text{cls}} + \mathcal{L}_{\text{reg}} + \mathcal{L}_{\text{g}}, \tag{10}$$

where we apply cross-entropy function for $\mathcal{L}_{\text{cls}}$ and smooth L1 function for $\mathcal{L}_{\text{reg}}$. The optimization problem:

$$\begin{aligned}\min_{\Theta} \mathcal{L}(\Theta) = \mathbf{1} \cdot \log\left[1 + \exp\left(-\boldsymbol{Y}_1 \hat{\mathbf{y}}_1\right)\right] \\ + \frac{1}{2}\left(\hat{\mathbf{y}}_2 - \mathbf{y}_2\right)^T \left(\hat{\mathbf{y}}_2 - \mathbf{y}_2\right) \\ + \frac{1}{2}\|\hat{\mathbf{y}}_1\|^2 - \frac{1}{2}\|\hat{\mathbf{y}}_2\|^2,\end{aligned} \tag{11}$$

where the last two terms are $\mathcal{L}_{\text{g}}$, $\boldsymbol{Y}_1 = \text{diag}(\mathbf{y}_1) \in \mathbb{R}^{n\times n}$ is the diagonal matrix of ground truth classification labels $\mathbf{y}_1 \in \mathbb{R}^n$, and $\mathbf{y}_2 \in \mathbb{R}^n$ is the ground truth regression labels [3]*. $\mathbf{1}$ denotes the all-ones vector with size n. We only consider the interval $[-1, 1]$ for smooth L1 function* [4]*. The above optimization problem can be transferred to the following maximization problem on the dual variable:*

$$\min_{\Theta} \mathcal{L}(\Theta) = \max_{\Phi} \mathcal{H}(\Phi), \tag{12}$$

$$\begin{aligned}\mathcal{H}(\Phi) = -\mathbf{1} \cdot \left[\Phi \log \Phi + (1-\Phi)\log(1-\Phi)\right] \\ - \Phi^T \boldsymbol{Y}_1 \psi \Delta w_1 + \frac{1}{2}\left(\psi \Delta w_2 - \mathbf{y}_2\right)^T \left(\psi \Delta w_2 - \mathbf{y}_2\right) \\ + \frac{1}{2}\|\psi \Delta w_1\|^2 - \frac{1}{2}\|\psi \Delta w_2\|^2,\end{aligned} \tag{13}$$

$$\Delta = \psi^{-1}\left(\frac{1}{w_1}\boldsymbol{Y}_1^T \Phi + \frac{w_2}{w_1^2}\mathbf{y}_2\right), \tag{14}$$

where $\Phi \in (0,1)^n$ is a variational parameter defined for each training example. Together with Proposition 1, the gradient descent for Φ is calculated as follows:

$$\frac{\partial \mathcal{H}(\Phi)}{\partial \Phi} = \log\frac{1-\Phi}{\Phi} - \boldsymbol{Y}_1 \boldsymbol{Y}_1^T \Phi - \frac{w_2}{w_1}\boldsymbol{Y}_1 \mathbf{y}_2. \tag{15}$$

The proof of Theorem 1 is shown as follows:

Proof. Recall the optimization problem:

$$\begin{aligned}\min_{\Theta} \mathcal{L}(\Theta) = \mathbf{1} \cdot \log\left[1 + \exp\left(-\boldsymbol{Y}_1 \hat{\mathbf{y}}_1\right)\right] \\ + \frac{1}{2}\left(\hat{\mathbf{y}}_2 - \mathbf{y}_2\right)^T \left(\hat{\mathbf{y}}_2 - \mathbf{y}_2\right) \\ + \frac{1}{2}\|\hat{\mathbf{y}}_1\|^2 - \frac{1}{2}\|\hat{\mathbf{y}}_2\|^2,\end{aligned} \tag{17}$$

According to (Jaakkola and Haussler 1999), we have the following inequality:

$$\begin{aligned}\log\left[1 + \exp\left(-\boldsymbol{Y}_1 \hat{\mathbf{y}}_1\right)\right] \geq -\left[\Phi \log \Phi + (1-\Phi)\log(1-\Phi)\right] \\ - \Phi \odot \boldsymbol{Y}_1 \hat{\mathbf{y}}_1,\end{aligned} \tag{18}$$

where $\odot$ is the element-wise vector product, and the equality holds when Φ achieves $\Phi^* = \frac{\partial \mathcal{L}}{\partial \boldsymbol{Y}_1 \hat{\mathbf{y}}_1}$ (Pezeshki et al. 2021). The minimization problem of $\mathcal{L}$ w.r.t Θ can be written as:

$$\min_{\Theta} \mathcal{L}(\Theta) = \min_{\Theta} \max_{\Phi} \mathcal{H}(\Phi, \Theta), \tag{19}$$

$$\begin{aligned}\mathcal{H}(\Phi, \Theta) = -\mathbf{1} \cdot \left[\Phi \log \Phi + (1-\Phi)\log(1-\Phi)\right] \\ - \Phi^T \boldsymbol{Y}_1 \hat{\mathbf{y}}_1 + \frac{1}{2}\left(\hat{\mathbf{y}}_2 - \mathbf{y}_2\right)^T \left(\hat{\mathbf{y}}_2 - \mathbf{y}_2\right) \\ + \frac{1}{2}\|\hat{\mathbf{y}}_1\|^2 - \frac{1}{2}\|\hat{\mathbf{y}}_2\|^2.\end{aligned} \tag{20}$$

Note that $\min_\Theta$ and $\max_\Phi$ can be swapped according to Lemma 3 in (Jaakkola and Haussler 1999) and we have:

$$\min_{\Theta} \mathcal{L}(\Theta) = \max_{\Phi} \min_{\Theta} \mathcal{H}(\Phi, \Theta) \tag{21}$$

Together with Proposition 1, we have the solution Θ^* for the r.h.s.:

$$\frac{\partial \mathcal{H}(\Phi, \Theta)}{\partial \Theta}\Big|_{\Theta=\Theta^*} = 0, \tag{22}$$

$$\begin{aligned}\frac{\partial \mathcal{H}(\Phi, \Theta)}{\partial \Theta} &= -w_1 \psi^T \boldsymbol{Y}_1^T \Phi + w_2 \psi^T\left(\psi \Theta w_2 - \mathbf{y}_2\right) \\ &\quad + w_1^2 \psi^T \psi \Theta - w_2^2 \psi^T \psi \Theta \\ &= -w_1 \psi^T \boldsymbol{Y}_1^T \Phi + w_2^2 \psi^T \psi \Theta - w_2 \psi^T \mathbf{y}_2 \\ &\quad + w_1^2 \psi^T \psi \Theta - w_2^2 \psi^T \psi \Theta \\ &= -w_1 \psi^T \boldsymbol{Y}_1^T \Phi - w_2 \psi^T \mathbf{y}_2 + w_1^2 \psi^T \psi \Theta.\end{aligned} \tag{23}$$

Then we have:

$$\Theta^*(\Phi) = \psi^{-1}\left(\frac{1}{w_1}\boldsymbol{Y}_1^T \Phi + \frac{w_2}{w_1^2}\mathbf{y}_2\right), \tag{24}$$

$$\Delta = \Theta^*(\Phi), \frac{\partial \Delta}{\partial \Phi} = \frac{\psi^{-1}\boldsymbol{Y}_1^T}{w_1}. \tag{25}$$

[3] To simplify, we consider each image contains one bounding box and one can easily expand it to multiple bounding boxes following the proof of this theorem.

[4] We can easily use the normalization to constrain the input of smooth L1 function.

Therefore, Equation (19) is transferred to

$$\min_{\Theta} \mathcal{L}(\Theta) = \max_{\Phi} \mathcal{H}(\Phi), \tag{26}$$

$$\begin{aligned}\mathcal{H}(\Phi) = &-\mathbf{1} \cdot [\Phi \log \Phi + (1-\Phi)\log(1-\Phi)] \\ &- \Phi^T \boldsymbol{Y}_1 \psi \Delta w_1 + \frac{1}{2}\left(\psi \Delta w_2 - \mathbf{y}_2\right)^T \left(\psi \Delta w_2 - \mathbf{y}_2\right) \\ &+ \frac{1}{2}\|\psi \Delta w_1\|^2 - \frac{1}{2}\|\psi \Delta w_2\|^2.\end{aligned} \tag{27}$$

Together with Equation (25), we have the following deduction:

$$\begin{aligned}\frac{\partial \mathcal{H}(\Phi)}{\partial \Phi} &= \log(1-\Phi) - \log \Phi \\ &\quad - 2\boldsymbol{Y}_1 \boldsymbol{Y}_1^T \Phi - \frac{w_2}{w_1}\boldsymbol{Y}_1 \mathbf{y}_2 + \frac{w_2}{w_1}\boldsymbol{Y}_1 \left(\psi \Delta w_2 - \mathbf{y}_2\right) \\ &\quad + \boldsymbol{Y}_1 \psi \Delta w_1 - \frac{w_2}{w_1}\boldsymbol{Y}_1 \psi \Delta w_2 \\ \frac{\partial \mathcal{H}(\Phi)}{\partial \Phi} &= \log \frac{1-\Phi}{\Phi} - 2\boldsymbol{Y}_1 \boldsymbol{Y}_1^T \Phi - \frac{2w_2}{w_1}\boldsymbol{Y}_1 \mathbf{y}_2 \\ &\quad + w_1 \boldsymbol{Y}_1 \psi \psi^{-1}\left(\frac{1}{w_1}\boldsymbol{Y}_1^T \Phi + \frac{w_2}{w_1^2}\mathbf{y}_2\right) \\ \frac{\partial \mathcal{H}(\Phi)}{\partial \Phi} &= \log \frac{1-\Phi}{\Phi} - \boldsymbol{Y}_1 \boldsymbol{Y}_1^T \Phi - \frac{w_2}{w_1}\boldsymbol{Y}_1 \mathbf{y}_2.\end{aligned} \tag{28}$$

□

Datasets

The Daytime-Sunny, Daytime-Foggy, Dusk-Rainy, Night-Sunny and Night-Rainy are constructed using Berkeley Deep Drive 100K (BDD100K) (Yu et al. 2020), Cityscapes (Cordts et al. 2016), Foggy Cityscapes (Sakaridis, Dai, and Van Gool 2018), and Adverse-Weather datasets (Hassaballah et al. 2020). Particularly, the Daytime-Sunny domain contains 27,708 images in total (19,395 images for training and 8,313 images for testing) selected from the BDD100K dataset. Specifically, the Daytime-Foggy domain contains 3,775 images selected from Foggy Cityscapes and Adverse-Weather datasets. The Dusk-Rainy and Night-Rainy domains contain 3,501 and 2,494 images, respectively, rendered from the BDD100K dataset following Wu et al. (2021). The Night-Sunny domain contains 26,158 images selected from the BDD100K dataset. For consistency, we follow Wu and Deng (2022) and consider seven common categories, including bus, bike, car, motor, person, rider, and truck.

Further Experiments

We conduct an ablation study on the value of λ_{g} to understand how it affects the overall generalization performance and find the optimal setting for it. Table 5 lists the results of different values for λ_{g} on the four target domains. These results show that G-NAS achieves the optimal average generalization performance with up to 33.5% when λ_{g} is set to 1. This is consistent with the value of λ_{g} set in Theorem 1 and further validate Theorem 1.

Method	λ_{g}	D-F	D-R	N-S	N-R	Avg.
G-NAS	0	33.6	28.0	35.2	16.1	28.2
G-NAS	0.01	34.0	31.2	41.4	**17.5**	31.0
G-NAS	0.1	34.1	31.5	41.6	17.0	31.1
G-NAS	1.0	**36.4**	**35.1**	**45.0**	17.4	**33.5**
G-NAS	2.0	34.6	31.8	42.0	17.3	31.4
G-NAS	5.0	34.3	31.0	40.3	16.9	30.6
G-NAS	10.0	32.3	29.5	39.3	16.7	29.5

Table 5: **Results of different hyper-parameters.** D, F, R, N, and S represent Daytime, Foggy, Rainy, Night, and Sunny, respectively.

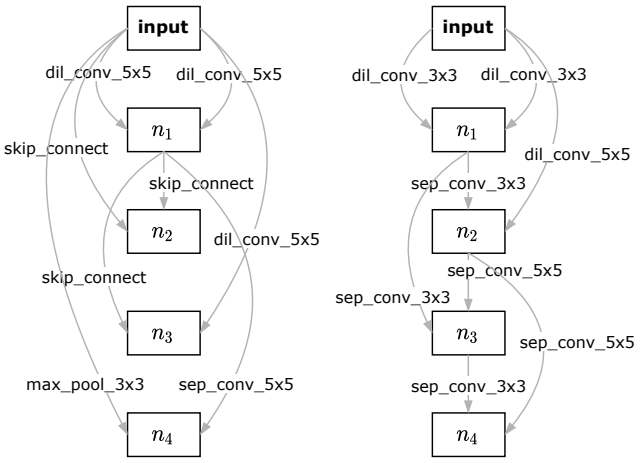

Figure 6: **Searched architectures of the normal cell (left) and reduction cell (right).** The searched cell contains four ordered nodes $\{n_1, n_2, n_3, n_4\}$ and each node has two previous inputs. Each directed edge denotes the chosen operation. The output of the cell is the concatenation of the output of each node.

Search Space

The design for the candidate operations is the same as DARTS (Liu, Simonyan, and Yang 2018), including average pooling with filter size 3×3, max pooling with filter size 3×3, separable convolutions with filter sizes 3×3, and 5×5, dilated separable convolutions with filter sizes 3×3, and 5×5, and skip connect. Note that our method is agnostic to the search space design and can be potentially extended to other search space designs. The search space mainly contains two types of cells—normal cells and reduction cells. Normal cells are foundational building blocks in our NAS framework. Reduction cells down-sample the input feature maps while normal cells maintain the size of feature maps after processing.

Searched Architectures

In this section, we visualize and analyze the searched architectures with the proposed method.

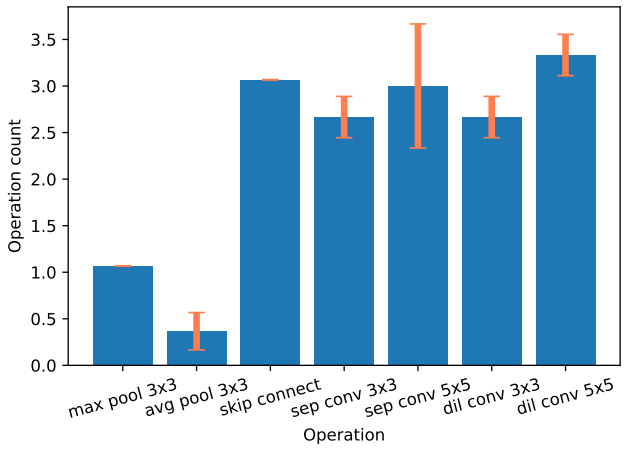

Figure 7: **Stability of searched architectures** with different initial random seeds. The operation percentage in each operation contains three bars, denoting the percentage of this operation in the searched architecture initialized by random seeds $0 \sim 2$ from left to right.

Patterns of searched architectures. The searched architecture is demonstrated in Figure 6. As shown in Figure 6 (left), the searched normal cells tend to contain more large-kernel convolution layers and skip layers. This enlarges the receptive field of the searched architecture and encourages the network to learn hard-to-learn global feature representations instead of only easy-to-learn local features. On the other hand, the reduction cell is used for down-sampling the input feature map. The reduction cell searched by G-NAS has similar numbers of large and small kernel convolutional layers, as shown in Figure 6 (right). This enables the reduction cell to simultaneously learn local features and global semantic features, preventing it from generating high-dimensional representations solely from local or global features. This further validates the motivation of G-NAS to improve generalization abilities.

Stability of searched architectures. We further test the stability of searched architectures against random initializations. The statistics of searched architectures with different random seeds are shown in Figure 7. As shown in the figure, the architectures searched by G-NAS converge to the pattern that the percentages of convolutional layer and skip connect are higher, while the pooling layer is lower. This results demonstrate the stability of the pattern found by G-NAS to improve OoD generalization ability.

Visualization Results

We visualize examples of inference results in Figure 8 and Figure 9. As shown in Figure 8 and Figure 9, the proposed method demonstrates robust object detection abilities under extremely challenging and unseen environments with only a single domain data for training. For example, vehicles are hardly recognized if covered by fog, while our proposed G-NAS accurately detect these vehicles compared with baselines, as shown in the first-row of Figure 8.

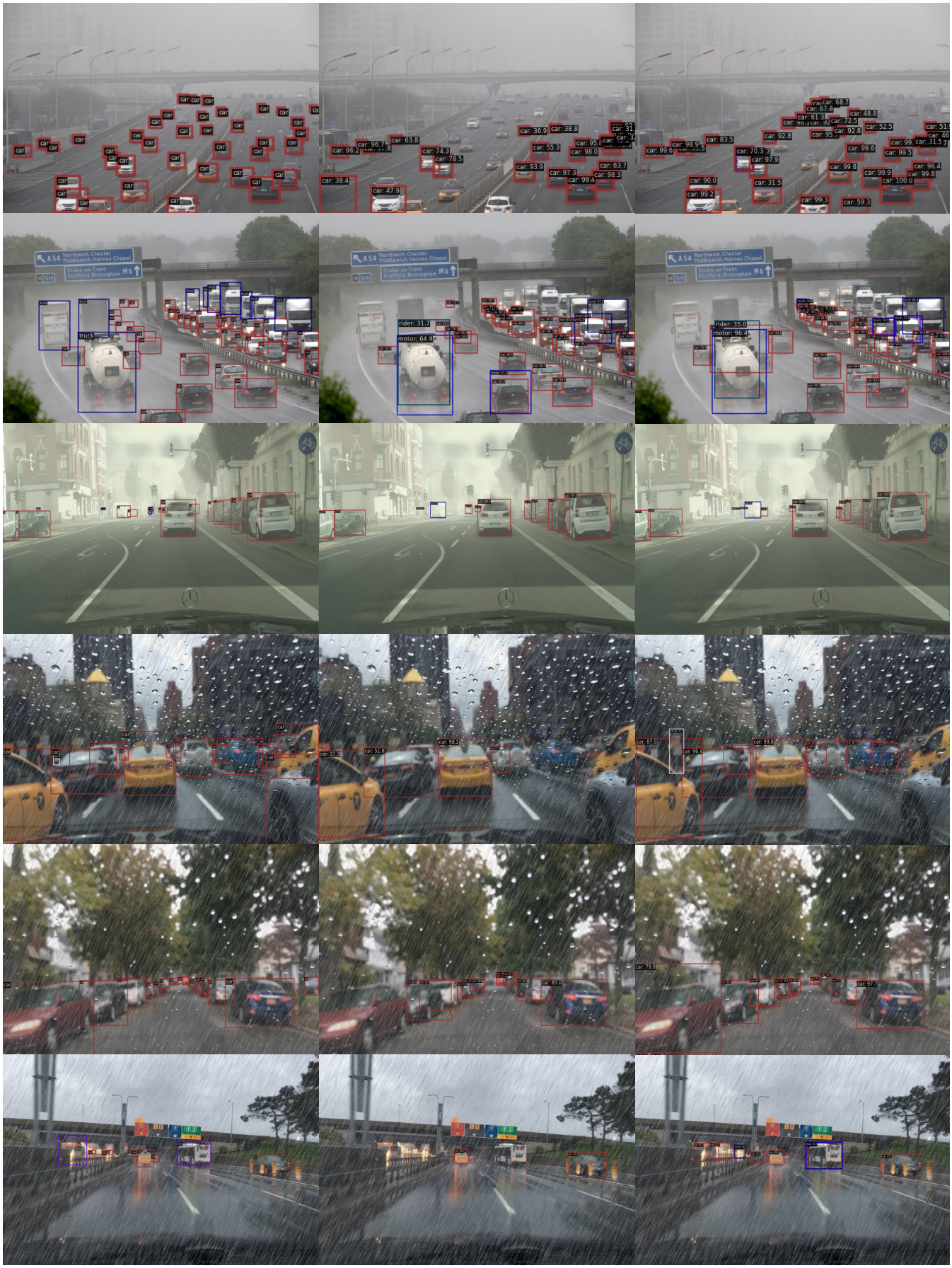

Figure 8: **Ground truth (left) and more inference results of Baselines (middle) and G-NAS (right).** The top three rows are on Daytime-Foggy and the bottom three rows are on Dusk-Rainy. The red boxes represent car objects, blue boxes represent truck objects, white boxes represent person objects and purple boxes represent bus objects.

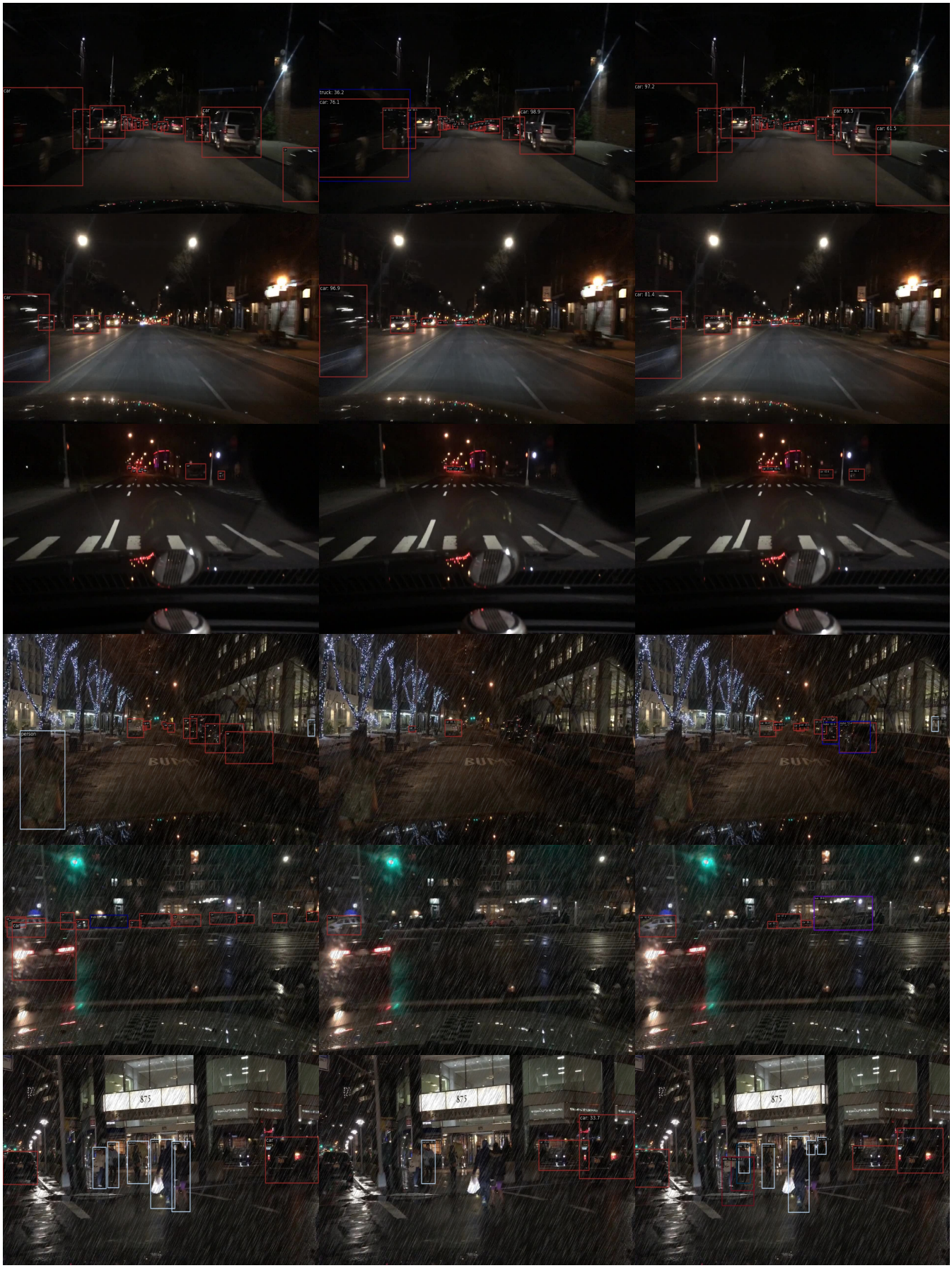

Figure 9: **Ground truth (left) and more inference results of Baselines (middle) and G-NAS (right).** The top three rows are on Night-Sunny and the bottom three rows are on Night-Rainy. The red boxes represent car objects, blue boxes represent truck objects, and white boxes represent person objects.