Fine-Tuning Strategies for Weather Condition Shifts: A Comparative Analysis of Models Trained on Synthetic and Real Datasets

Jungwoo Kim[1], Min Jung Lee[2], Suha Kwak[3,*]
[1,2]Graduate School of Artificial Intelligence, POSTECH
[3]Dept. of Computer Science, Graduate School of Artificial Intelligence, POSTECH

{jwk00216, minjlee, suha.kwak}@postech.ac.kr

Abstract

Despite advancements in deep learning, existing semantic segmentation models exhibit suboptimal performance under adverse weather conditions, such as fog or rain, whereas they perform well in clear weather conditions. To address this issue, much of the research has focused on making image or feature-level representations weather-independent. However, disentangling the style and content of images remains a challenge. In this work, we propose a novel fine-tuning method, 'freeze-n-update.' We identify a subset of model parameters that are weather-independent and demonstrate that by freezing these parameters and fine-tuning others, segmentation performance can be significantly improved. Experiments on a test dataset confirm both the effectiveness and practicality of our approach.

1. Introduction

Semantic segmentation models typically perform well in clear weather conditions. However, their performance significantly degrades under adverse weather conditions, such as rain or fog. Extensive research has focused on separating image or feature-level representations to handle this issue, but fully disentangling the style (e.g., fog and rain) from the image content remains difficult. To address this challenge, we introduce a novel fine-tuning method, freeze-n-update, aimed at improving semantic segmentation in adverse weather. We posit that not all model parameters are affected by weather conditions and demonstrate this using kernel wise and layer wise comparisons across multiple backbone architectures. Utilizing this property of parameters, we enhance segmentation performance. Both qualitative and quantitative results corroborate the validity of our assumption and the efficacy of the proposed fine-tuning method.

2. Method

In this section, we demonstrate the practicality of freezing a certain percentage of parameters. We first outline the training details, including the datasets and the existing model we adopted. Then, we delve into an in-depth analysis of the results from three different types of models. Finally, we present the details of the proposed new fine-tuning method, underscoring its novel features and advantages.

2.1 Training details

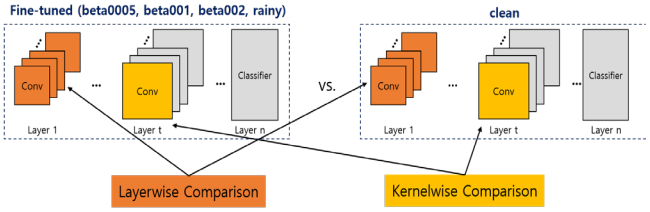
Dataset & Model: Due to poor visibility in adverse weather conditions, both models and humans struggle to obtain high-quality ground-truth labels. Moreover, we aim to identify which model parameters are sensitive to changes in weather. Therefore, we hypothesize that "only certain parameters are affected by weather changes." Our research requires a dataset with consistent content but varying weather conditions. We use the synthetic dataset Foggycityscapes[1], which adds a realistic fog effect to the fully annotated Cityscapes[2] dataset. Similarly, to assess the generalizability of our hypothesis, we use the Raincityscapes[3] dataset, which enables us to test our approach under different weather conditions, such as rain. We utilize DeepLabv3+[4], a stable and widely used model, to apply our new fine-tuning method. To verify the method's generalizability, we analyze parameters using two different backbone networks, ResNet101[5] and MobileNet[6], which allows us to assess effectiveness across varying architectures.

Training details: In the Foggycityscapes dataset, the beta value determines the amount of synthesized fog, with a higher beta value resulting in thicker fog. Here, three beta values are given: [0.005, 0.01, 0.02]. We fine-tune all parameters of the model for each beta value to observe changes as the fog density increases. Additionally, we fine-tune the model using the Raincityscapes dataset. For convenience, we will refer to the models fine-tuned at each beta level as foggy-beta0005, foggy-beta001, foggy-beta002, and the model fine-tuned for rain as rainy. Finally, we train the model on a clear weather Cityscapes dataset to serve as a reference model for comparison. This model will be referred to as clean.

2.2 Analysis parameters of trained models

We analyze how many parameters are shared among the different fine-tuned models through two types of comparisons:

kernel wise and layer wise. For example, in a convolutional layer with parameters defined by (out, in, kernel size) = (32, 3, 3 x 3), the parameter tensor's shape is (32, 3, 3, 3). Kernel wise comparison involves comparing each individual kernel, with a shape of (3, 3, 3), to the corresponding kernel in another model at the same location. Layer wise comparison involves comparing the entire parameter tensor of one layer, also shaped (32, 3, 3, 3), to the equivalent layer in another model. The entire process is illustrated in Fig. 1.



(Figure 1) Illustration of our analysis method

With Normalization: Before comparing the parameters, we normalize all parameters to follow a normal distribution to handle outliers more effectively. This involves subtracting the mean of the parameter values and dividing by their standard deviation (Eq. 1).

$$P_{t,c,n,normalized} = \frac{P_{t,c,n} - \mu_{t,c}}{\sigma_{t,c}}, \quad (1)$$

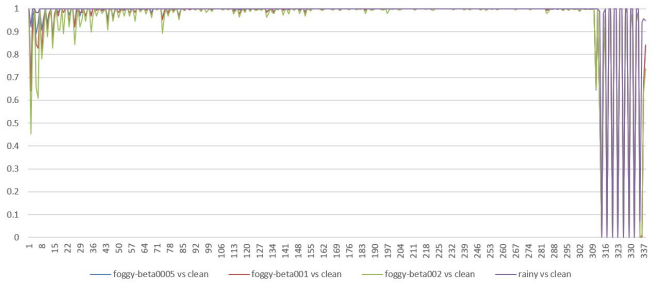
Where P represents a tensor of parameters, and t denotes the dataset type, which includes {clean, foggy-beta0005, foggy-beta001, foggy-beta002, rainy}. The variable c indicates the comparison method, either {layerwise, kernelwise}, and n is the index of the tensor. The term 'normalized' refers to parameters adjusted to follow a normal distribution, where μ is the mean and σ is the standard deviation of the parameter distribution. Using these normalized parameters, we calculate the number of differing parameters between the models trained on the clean Cityscapes and those trained on the foggy or rainy Cityscapes. This calculation is performed according to Eq. 2.

$$X_{t,c,n,normalized} = \begin{cases} 1, & \text{if } \left| P_{t,c,n,normalized} - P_{clean,c,n,normalized} \right| < T \\ 0, & \text{otherwise} \end{cases}, \quad (2)$$

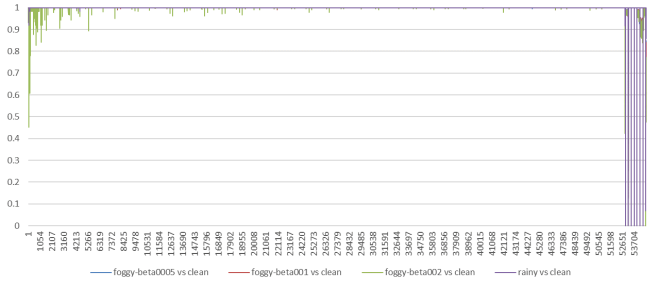
Here, X is a tensor representing the differences at the same positions between parameters of fine-tuned models and the clean model. Each element of X is set to 1 if the difference is smaller than a specified threshold, indicating similarity, and 0 otherwise. Thus, X identifies where elements of the fine-tuned models closely match those of the clean model.

T represents the threshold value used in our analysis. We set T = 0.01 for models using ResNet101 as a backbone and T = 0.1 for those using MobileNet. These thresholds are adjusted according to the different learning rates of the backbones to ensure that the comparison results are consistent across models. By applying these thresholds in Equation 2, we compare the parameters of the fine-tuned models to those of the clean model, identifying where the parameters have not significantly deviated. we calculate how many '1's are in the tensor and then divide the sum by the total number of elements in order to get a ratio of the tensor.

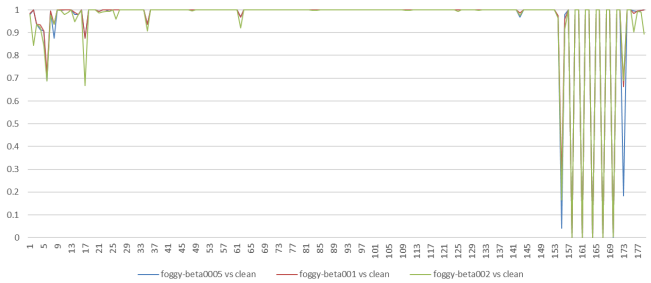
Analysis result: Figure 2~5 shows the comparison results of models over two types of backbone, ResNet101 and MobileNet, via two comparison methods. The ratio of classifier layers which is the rightmost part of four graphs is much lower than others. This leads us to conclude parameters in classifier layers are affected by weather conditions more than the parameters in the other layers. Moreover, the shallow part of layers, accurately convolution layers and batchnormalization layers are also affected by the weather conditions.



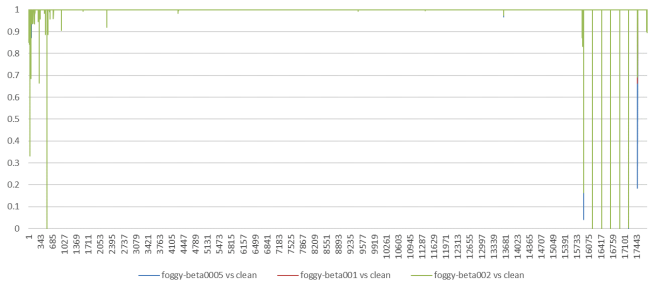
(Figure 2) Layer wise comparison result in ResNet101



(Figure 3) Kernel wise comparison result in ResNet101



(Figure 4) Layer wise comparison result in MobileNet



(Figure 5) Kernel wise comparison result in MobileNet

2.3 New finetuning method

Based on the results of our parameter analysis, we identified specific layers that react sensitively to changes in weather conditions. We focused on the top 50 layers showing the largest variations in parameters for each beta value under foggy and rainy conditions. Our comparison sought to distinguish between layers with consistent changes and those with significant fluctuations. We found that the layers most affected were the front-end convolutional layers, the batch normalization layers (BN) across the entire model, and the classifier section at the model's end. Consequently, we propose fine-tuning only the first 10 convolutional layers along with the BN and classifier sections in ResNet101, while freezing the remaining parameters. Should the backbone network change, the number of convolutional layers adjusted for fine-tuning should be modified accordingly.

3. Experiments

We use SGD as the optimizer with DeepLabv3+ as the model framework. The learning rate is set to 0.1 for MobileNet and 0.01 for ResNet, respectively. Additionally, we adopt a polynomial learning rate decay with a power of 0.9. During the training process, the input images are resized to a 768 x 768 size and randomly flipped horizontally. We set the batch size to 16 and the output stride to 16. All experiments are conducted using the same hyperparameters as mentioned above. Furthermore, we test the trained models on the real-world Foggy driving[1] dataset to evaluate their performance under practical conditions. We used the mean of class-wise Intersection over Union (mIoU) as the evaluation metric.

3.1 Results

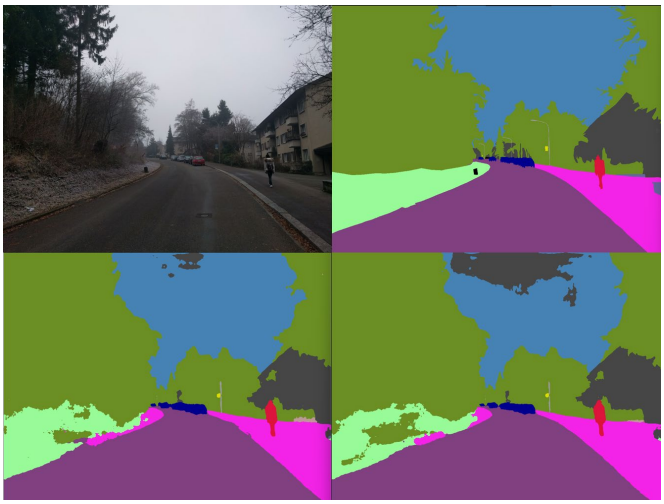
The performance of our proposed fine-tuning method is illustrated in Table 1. In the table, "CV" denotes the convolutional layers at the front of the model, while "CL" stands for the classifier. Generally, adapting a model to different weather conditions involves methods such as fine-tuning only the classifier or the entire model. However, our fine-tuning method outperforms both of these approaches. These results suggest that focusing on a subset of parameters sensitive to weather conditions is more effective than simply retraining the entire model. Additionally, while there is a method that fine-tunes only the front part of the model,[7] our approach—which includes learning both the batch normalization (BN) and the classifier sections along with the front part—achieves higher performance.

<Table 1> Performance comparison of fine-tuning methods

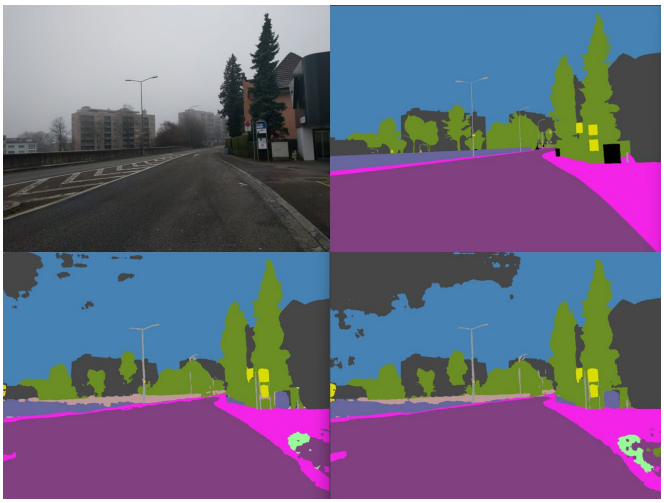
Method	CV	CL	All Layer	CV+CL	**CV + BN+ CL(our method)**
Mean IoU	0.4154	0.4182	0.4239	0.424494	**0.4304**
Mean Accuracy	0.5665	0.5751	0.5901	0.5811	**0.5910**

Our method offers numerous benefits, even though it utilizes far fewer parameters than full-model fine-tuning. By employing this technique, memory usage is optimized as most parameters remain frozen. This method also enables the model to adapt to various weather conditions by altering only a small set of parameters. Previously, we verified that only some parameters undergo significant changes under both rainy and foggy conditions.

The difference between the prediction results of traditional methods and our method is depicted in Figure 6 and Figure 7, where the top left image is the input image, and the top right image represents the ground truth labeling. On the bottom row, the left image shows the prediction by our fine-tuned model, while the right image depicts the prediction of a model with full fine-tuning. While existing methods struggle to recognize the sky, a class highly sensitive to weather changes, in foggy conditions, our model accurately identifies it even under such challenging conditions.



(Figure 6) Visualization of Segmentation Prediction



(Figure 7)

4. Conclusion

We have proposed a new fine-tuning method for vision recognition tasks in adverse weather conditions. Additionally, by comparing parameters of models trained in both clean and adverse weather, we have identified which parameters or layers are affected. Our study can be helpful in developing other fine-tuning methods or in studying domain adaptation

methods. Also, our proposed method demonstrates better performance than existing methods. These results suggest that utilizing the identified parameter characteristics can be beneficial.

References

[1] Sakaridis, Christos, Dengxin Dai, and Luc Van Gool. "Semantic foggy scene understanding with synthetic data" International Journal of Computer Vision 126, pp. 973-992, 2018.

[2] Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., ... & Schiele, B. “The cityscapes dataset for semantic urban scene understanding” In Proceedings of the IEEE conference on computer vision and pattern recognition, San Diego, 2016, pp. 3213-3223.

[3] Hu, X., Fu, C. W., Zhu, L., & Heng, P. A. “Depth-attentional features for single-image rain removal.” In Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition, California, 2019, pp. 8022-8031.

[4] Chen, L. C., Zhu, Y., Papandreou, G., Schroff, F., & Adam, H. “Encoder-decoder with atrous separable convolution for semantic image segmentation” In Proceedings of the European conference on computer vision (ECCV), Munich, 2018, pp. 801-818.

[5] He, K., Zhang, X., Ren, S., & Sun, J. “Deep residual learning for image recognition” In Proceedings of the IEEE conference on computer vision and pattern recognition, Las Vegas, 2016, pp. 770-778.

[6] Howard, Andrew G., et al. “Mobilenets: Efficient convolutional neural networks for mobile vision applications.” arXiv preprint arXiv:1704.04861, 2017.

[7] Lee, Y., Chen, A.S., Tajwar, F., Kumar, A., Yao, H., Liang, P., Finn, C. “Surgical fine-tuning improves adaptation to distribution shifts”, ICLR, 2022.