Supplementary Materials for On-the-fly Category Discovery for LiDAR Semantic Segmentation

Hyeonseong Kim*, Sung-Hoon Yoon*, Minseok Kim, and Kuk-Jin Yoon

Visual Intelligence Lab., KAIST, Korea
{brian617,yoon307,alstjrx1x1,kjyoon}@kaist.ac.kr

We provide the following contents in the supplementary material:

- Hyperparameter analysis.
- Class-wise results.
- Details of the proposed method, baselines, and evaluation protocol.
- Additional qualitative results.

1 Hyperparameter Analysis

We analyze the effects of hash code dimension L, mixing ratio r, weight λ, and sub-sampling in the grouping module (GM). Due to the extensive nature of the experiments on hyperparameters, we conduct experiments on POSS_1 split in *scenario-A*.

Table 1: Results according to hash code dimension L.

	Strict-Hungarian (%)			Greedy-Hungarian (%)		
L	Unknown	Known	All	Unknown	Known	All
8	22.90	44.01	39.14	30.23	49.20	44.82
10	**27.09**	43.67	39.84	31.78	47.87	44.15
12	26.68	44.11	40.09	**39.28**	**50.47**	**47.89**
14	25.91	45.05	**40.63**	25.27	49.84	44.17
16	17.63	**45.25**	38.88	20.07	48.60	42.02

Hash code dimension L. In Tab. 1, we conduct an experiment to demonstrate the performance variations according to the hash code dimension L. When the dimension is L, the number of hash codes that can be expressed is 2^L, where smaller L effectively handles intra-class variation, while a model with larger L has high representation capability [2]. To demonstrate the resilience of our method to the hash code dimension, we vary the value of L from 8 to 16. Our method exhibits robust performance for the known classes regardless of the code dimension. Considering the number of classes in the LiDAR dataset, where SemanticPOSS has 13 classes and SemanticKITTI has 19 classes, it can be seen that a sufficient number of hash codes are being created to properly represent each class within the given search space ($L \in \{8, 10, 12, 14, 16\}$). However, when the hash code

* Equal contribution.

dimension is small (*i.e.* L=8), there is a decrease in the expressive power for unknown classes, resulting in slightly lower performance. Conversely, when the dimension is too large (*i.e.* L=16), the space allocated for unknown classes becomes extensive, creating meaningless representations and a subsequent decrease in performance for unknown classes. Nonetheless, our method demonstrates robust performance in both protocols within the search space, except when the dimension is too large.

Table 2: Results with different mixing ratios. Fixed (r) refers to using the fixed mixing ratio of r. Beta (α, β) refers to sampling the mixing ratio r from the Beta distribution with parameters α and β at each training iteration.

		Strict-Hungarian (%)			Greedy-Hungarian (%)		
	Params	Unknown	Known	All	Unknown	Known	All
Fixed (r)	0.05	22.89	**44.77**	39.72	28.07	49.74	44.74
	0.1	26.68	44.11	**40.09**	**39.28**	**50.47**	**47.89**
	0.25	17.58	42.76	36.95	26.71	49.35	44.13
	0.5	21.31	39.38	35.21	23.70	48.55	42.81
Beta (α, β)	(0.5, 0.5)	25.22	44.02	39.68	29.53	49.65	45.00
	(1, 1)	**28.43**	39.58	37.00	31.69	44.24	41.35
	(2, 2)	18.87	40.37	35.41	22.98	44.65	39.65

Mixing ratio r. To show the effects of mixing strategy in modeling the out-of-distribution (OOD) representation, we experiment with two different mixing strategies: 1) fixed mixing ratio and 2) sampling the mixing ratio from Beta distribution. The results are shown in Tab. 2. For the fixed mixing ratio, we fix the ratio r when mixing representations. When the ratio is small (*i.e.* r=0.05), the mixed representation is close to the in-distribution (ID) representations, resulting in high performance for the known classes. However, since the OOD representations are modeled close to ID representations, the performance improvement for unknown classes is slightly reduced compared to r=0.1. Conversely, when mixing the representations with an average ratio (*i.e.* r=0.5), OOD representations are well modeled, but the performance improvement is slightly reduced compared to r=0.1 due to low feasibility. In the case of r=0.1, performances for both known and unknown classes are high, indicating feasible OOD representations are modeled. Beyond the fixed mixing ratio, we also use a strategy that samples mixing ratio r from Beta distribution as proposed in [8, 9]. Specifically, we sample the mixing ratio at each training iteration and use it within mini-batches. Consistent with the results above, mixing representations from the adjacent regions of ID representations (α,β=0.5) yields the best overall performance. On the other hand, when more representations are mixed at distant space (α,β=2.0), performances of both known and unknown classes are lower than in other cases. Nevertheless, all mixing strategies exhibit satisfactory performances, demonstrating the proposed method applies to both mixing strategies. While we use a simple yet effective fixed mixing ratio strategy in this work, we believe

that developing an effective OOD modeling in the context of OCDSS would be valuable for future research.

Table 3: Results according to weight λ in grouping module.

	Strict-Hungarian (%)			Greedy-Hungarian (%)		
λ	Unknown	Known	All	Unknown	Known	All
0.001	17.12	**44.80**	38.41	26.02	**50.74**	45.03
0.01	24.41	44.44	39.81	34.04	48.91	45.48
0.05	**26.68**	44.11	**40.09**	**39.28**	50.47	**47.89**
0.1	25.72	38.82	35.80	27.54	42.63	39.15
0.2	20.18	37.57	33.56	24.90	41.13	37.39

Weight λ. To observe the effect of loss weight λ for GM, we experiment by varying λ as shown in Tab. 3. As λ increases, it can be seen that the performance for unknown classes gradually improves through learning the grouping ability for unknown classes from GM. However, when λ exceeds a certain level ($\lambda > 0.1$), we can observe a degradation in the performance of the known classes. Additionally, the performance for unknown classes saturates with a slight decrease, as the model focuses on learning to cognize rather than to recognize. For a large value of λ (*i.e.* $\lambda = 0.2$), we speculate that the loss of recognition ability may also impair cognition ability. Our experimental results affirm that the proposed method exhibits robust and satisfactory performance within the suitable range of λ (0.01$\sim$0.1).

Table 4: Results with different number of sampling in grouping module.

	Strict-Hungarian (%)			Greedy-Hungarian (%)		
# of Sampling	Unknown	Known	All	Unknown	Known	All
512	**28.11**	43.14	39.67	35.01	48.19	45.15
1024	26.68	44.11	40.09	**39.28**	**50.47**	**47.89**
2048	26.76	**44.79**	**40.63**	34.56	49.69	46.20

Sub-sampling in GM. We perform grouping between mixed representations in GM, which requires pair-wise computation. Since each scan in the LiDAR data contains a large number of points, utilizing all the points for pair-wise computation in training increases the memory consumption and computational burden. To alleviate the burden, we sample a fixed number of points for the computation. In Tab. 4, we show the performance of using different numbers of sampled points. The proposed method consistently shows satisfactory performances in all three cases. Increasing the number of points slightly enhances overall performance, but as meaningful performance is demonstrated even with using 1024 points, we opted to sample 1024 points in the GM.

Table 5: On-the-fly category discovery results on SemanticPOSS within *scenario-A*. Oracle method is trained with all classes. The values highlighted in gray are observed-unknown classes in each split. The best value is in **bold**.

	Split	Method	person	rider	car	trunk	plants	traffic.	pole	trashc.	build.	cone/stone	fence	bike	ground	mIoU Unknown	mIoU Known	mIoU All
		Oracle	56.39	56.88	40.70	44.98	75.58	33.35	16.05	11.26	73.22	27.09	34.45	50.15	79.90	-	46.16	46.16
Strict-Hungarian (%)	$POSS_0$	Baseline	25.37	24.30	25.02	**38.04**	27.94	1.91	6.99	1.14	52.66	26.36	25.93	5.82	32.67	9.20	28.59	22.63
		MLDG	21.45	54.31	30.25	21.84	17.85	1.43	15.85	0.50	47.26	27.17	39.87	5.54	54.20	6.33	34.69	25.96
		SMILE	44.77	47.89	42.78	24.06	10.78	5.05	**20.89**	0.94	67.60	19.07	25.47	**6.40**	70.51	5.79	40.34	29.71
		Ours	**63.72**	**55.72**	**43.02**	23.84	**57.69**	**6.16**	17.81	**2.53**	**74.38**	**38.14**	**52.00**	5.16	**77.77**	**17.88**	**49.60**	**39.84**
	$POSS_1$	Baseline	14.34	38.93	14.08	26.13	14.39	20.50	6.58	0.44	18.71	35.03	43.16	3.22	35.74	13.21	23.16	20.87
		MLDG	7.72	**44.58**	25.80	30.16	57.43	19.27	**16.51**	2.18	22.93	23.10	31.47	30.07	25.60	15.72	28.97	25.91
		SMILE	16.97	39.99	25.85	34.08	**65.65**	27.58	13.53	3.30	15.10	34.34	26.69	0.87	66.36	15.20	32.47	28.48
		Ours	**18.92**	44.49	**42.07**	**42.30**	61.83	**29.60**	3.77	**7.44**	**57.34**	**38.98**	**49.62**	**52.81**	**71.98**	**26.68**	**44.11**	**40.09**
	$POSS_2$	Baseline	38.00	**14.01**	24.32	26.08	25.55	17.33	12.65	0.24	21.32	3.43	**26.51**	29.26	22.57	21.03	19.82	20.10
		MLDG	30.28	6.33	22.79	**43.63**	47.57	21.40	13.84	**15.57**	62.04	4.42	7.49	40.17	16.31	10.04	30.17	25.52
		SMILE	27.51	13.50	44.64	42.56	60.47	18.19	9.06	1.62	**78.52**	**33.59**	4.65	0.24	8.83	9.00	31.64	26.42
		Ours	**52.83**	8.82	**56.34**	24.98	**72.47**	**30.41**	**24.58**	4.08	64.15	32.70	10.01	**51.48**	**54.73**	**24.52**	**41.40**	**37.51**
	$POSS_3$	Baseline	26.48	20.82	2.80	**9.00**	37.80	16.72	20.73	8.15	37.02	1.61	**45.59**	30.22	45.70	4.47	28.92	23.28
		MLDG	19.05	54.12	2.79	2.68	47.49	25.51	16.94	7.96	39.36	1.23	31.45	25.56	63.78	2.24	33.12	26.00
		SMILE	57.19	52.03	1.91	1.49	51.70	7.89	22.21	**12.19**	79.48	**2.28**	37.75	44.09	1.44	1.89	36.60	28.59
		Ours	**59.69**	**60.93**	**48.53**	1.13	**75.63**	**30.13**	**26.36**	4.04	**80.59**	1.60	45.27	**54.86**	**76.43**	**17.09**	**51.39**	**43.48**
															Avg Baseline	11.98	25.12	21.72
															Avg MLDG	8.58	31.74	25.85
															Avg SMILE	7.97	35.26	28.30
															Avg Ours	**21.54**	**46.63**	**40.36**
Greedy-Hungarian (%)	$POSS_0$	Baseline	27.17	24.31	36.40	42.49	29.94	3.89	15.44	1.76	55.58	30.54	27.58	**10.03**	34.87	11.41	32.71	26.16
		MLDG	25.13	59.04	40.82	56.60	18.79	1.60	**22.11**	0.78	51.11	2.29	53.19	9.04	54.47	7.55	40.54	30.39
		SMILE	60.50	48.04	58.45	**74.76**	16.56	6.14	19.62	3.62	**86.33**	19.32	58.62	9.86	**83.07**	9.05	**56.62**	41.91
		Ours	**70.70**	**59.19**	**68.72**	26.08	**59.30**	**8.89**	21.51	**5.15**	79.83	**39.30**	**62.93**	7.89	80.62	**20.30**	56.54	**45.39**
	$POSS_1$	Baseline	**24.01**	**55.66**	28.47	27.27	14.57	27.13	10.43	3.90	25.60	35.30	47.95	25.06	35.80	20.01	30.11	27.78
		MLDG	15.34	49.56	40.49	38.35	62.83	**48.29**	24.16	0.72	31.47	23.57	34.31	32.46	25.95	23.66	35.65	32.89
		SMILE	18.99	50.07	54.65	**48.58**	**72.75**	41.86	14.94	**5.87**	20.73	39.20	45.77	30.62	**74.60**	18.22	46.40	39.89
		Ours	23.02	52.42	**60.85**	46.73	64.87	46.15	**35.33**	5.57	**59.50**	**39.90**	**57.00**	**56.99**	74.19	**39.28**	**50.47**	**47.89**
	$POSS_2$	Baseline	49.25	30.96	31.35	26.28	26.07	17.40	13.16	2.19	22.28	31.01	**29.43**	52.50	25.14	28.51	27.15	27.46
		MLDG	38.15	29.28	29.27	43.95	51.92	21.58	13.87	**16.94**	63.68	9.72	10.38	44.40	19.79	19.82	33.40	30.27
		SMILE	37.19	18.39	**62.17**	**44.00**	78.05	18.61	9.12	4.03	**80.17**	**45.51**	10.71	3.53	46.70	25.27	38.24	35.24
		Ours	**73.15**	**52.83**	59.72	25.92	**82.83**	**31.78**	**24.85**	5.68	64.50	37.95	10.73	**60.75**	**55.22**	**39.59**	**46.71**	**45.07**
	$POSS_3$	Baseline	27.58	20.85	23.38	22.94	38.83	16.49	22.17	0.31	37.05	8.06	**46.41**	30.86	45.75	18.12	28.63	26.21
		MLDG	21.05	54.23	8.82	28.10	48.95	**32.75**	16.97	2.46	39.48	9.36	33.64	26.12	63.93	15.43	33.96	29.68
		SMILE	58.43	52.11	8.38	54.68	54.32	8.26	22.25	**13.62**	80.08	9.05	38.87	44.37	1.45	24.04	37.38	34.30
		Ours	**64.50**	**61.12**	**57.31**	**72.39**	**79.46**	31.05	**26.95**	3.59	**81.02**	**17.87**	45.48	**55.22**	**77.02**	**49.19**	**52.54**	**51.77**
															Avg Baseline	19.51	29.65	26.90
															Avg MLDG	19.15	35.89	30.81
															Avg SMILE	16.62	44.64	37.84
															Avg Ours	**37.09**	**51.57**	**47.53**

Table 6: On-the-fly category discovery results on SemanticKITTI within *scenario-A*. Oracle method is trained with all classes. The values highlighted in gray are observed-unknown classes in each split. The best value is in **bold**.

	Split	Method	car	bi.cle	mt.cle	truck	oth-v.	pers.	bi.clst	mt.clst	road	park.	sidew.	oth-g	build.	fence	veget.	trunk	terra.	pole	traff.	mIoU Unknown	mIoU Known	mIoU All
		Oracle	92.31	6.31	26.25	65.73	42.28	32.32	52.15	0.02	90.99	41.01	75.65	2.02	88.01	46.54	82.51	54.70	72.58	39.82	24.09	-	49.23	49.23
Strict-Hungarian (%)	$KITTI_0$	Baseline	39.79	0.44	20.73	**54.61**	6.31	16.67	33.91	1.37	44.36	15.81	21.82	**0.89**	47.73	8.13	20.27	5.46	19.85	24.12	24.88	10.47	25.34	21.43
		MLDG	58.53	0.23	2.02	25.04	7.97	1.09	28.55	**2.07**	31.93	13.39	35.86	0.52	48.51	20.86	19.98	14.06	21.01	28.86	27.12	12.65	23.17	20.40
		SMILE	85.77	1.06	7.11	25.51	**9.61**	28.94	51.70	0.51	44.14	0.03	0.85	0.12	69.83	26.36	20.06	3.06	15.89	24.29	28.08	9.94	28.09	23.31
		Ours	**88.19**	**2.05**	**23.97**	48.16	3.10	**35.49**	**56.21**	0.02	**87.95**	**26.14**	**59.94**	0.71	**81.36**	**48.89**	**63.09**	**16.25**	**28.26**	**52.17**	**33.57**	**22.55**	**45.91**	**39.77**
	$KITTI_1$	Baseline	45.86	0.39	21.08	25.48	19.47	27.65	**27.55**	**3.15**	12.41	4.90	27.59	0.79	52.11	**16.74**	14.12	43.79	22.29	24.25	**10.09**	13.51	23.72	21.04
		MLDG	65.38	**0.47**	23.91	32.32	7.85	29.49	15.42	1.96	11.87	12.09	19.62	**1.57**	37.73	11.34	54.79	36.47	35.09	**47.02**	4.72	8.99	28.87	23.64
		SMILE	**89.39**	0.04	**25.33**	**47.74**	23.54	**55.78**	9.02	0.27	17.38	9.93	0.93	0.47	62.33	5.07	**75.59**	47.23	38.72	35.75	0.98	6.59	36.61	28.71
		Ours	59.60	0.35	13.25	15.90	**33.07**	49.68	24.51	0.08	**73.42**	**21.94**	**67.32**	0.95	**81.42**	3.17	48.25	**62.20**	**67.69**	21.02	9.11	**22.23**	**38.70**	**34.36**
	$KITTI_2$	Baseline	16.01	0.24	0.35	**31.35**	5.38	22.27	**60.12**	**2.58**	39.03	19.63	11.64	0.67	38.18	20.42	44.53	34.35	39.86	**19.58**	18.48	10.03	26.75	22.35
		MLDG	12.13	2.28	0.54	22.82	9.24	20.66	36.84	1.87	70.69	11.81	17.18	0.38	61.74	26.85	56.89	32.26	42.92	15.75	26.78	9.50	30.15	24.72
		SMILE	**45.05**	0.25	0.88	29.92	2.51	**44.80**	47.64	0.71	51.70	26.09	2.55	0.11	**74.19**	3.68	72.33	37.07	24.53	6.40	20.49	11.12	31.09	25.84
		Ours	22.14	**3.15**	**1.70**	0.10	**27.23**	26.32	34.05	0.06	**87.31**	**32.97**	**33.44**	**0.85**	64.71	**37.42**	**75.69**	**43.92**	**69.21**	17.57	**34.59**	**14.98**	**38.39**	**32.23**
	$KITTI_3$	Baseline	69.96	**4.11**	22.19	26.45	6.52	1.34	52.62	0.05	28.03	8.17	53.06	**0.75**	15.75	22.45	24.31	25.34	18.70	**45.93**	17.78	12.93	26.12	23.34
		MLDG	70.24	2.46	0.67	3.35	17.96	1.44	46.94	0.66	48.04	**9.45**	28.01	0.47	16.98	10.15	41.32	27.58	41.12	22.57	23.36	7.80	25.44	21.72
		SMILE	90.71	1.73	**34.66**	**32.69**	15.77	0.74	42.11	0.07	**84.70**	5.11	43.43	0.11	38.31	0.63	55.48	40.96	9.59	21.53	23.08	**19.21**	30.97	28.49
		Ours	**92.20**	0.23	29.31	6.36	**22.49**	**7.47**	**63.58**	**3.63**	76.40	7.66	**67.42**	0.27	**49.50**	**49.54**	**76.85**	**57.47**	**65.68**	42.67	**29.63**	17.74	**45.15**	**39.38**
																				Avg	Baseline	11.74	25.48	22.04
																					MLDG	9.74	26.91	22.62
																					SMILE	11.72	31.69	26.59
																					Ours	**19.38**	**42.04**	**36.44**
Greedy-Hungarian (%)	$KITTI_0$	Baseline	40.19	5.84	36.41	65.14	12.70	24.81	34.56	2.54	44.36	16.05	22.13	1.31	54.47	24.24	20.49	5.60	21.49	24.75	33.00	13.22	30.28	25.79
		MLDG	59.96	1.06	1.47	60.50	28.71	2.76	28.62	**4.18**	31.93	4.54	44.06	**2.23**	50.07	34.35	21.13	15.08	36.98	30.76	30.35	20.59	27.34	25.56
		SMILE	88.56	1.39	48.68	63.40	25.33	11.53	53.50	4.15	57.84	0.05	3.53	1.53	**91.69**	**65.08**	20.32	3.17	**58.07**	24.99	30.97	21.66	38.96	34.41
		Ours	**96.00**	**8.26**	**53.27**	**70.46**	**36.52**	**58.02**	**65.51**	0.03	**88.93**	**27.13**	**64.68**	0.76	84.49	51.60	**66.40**	**17.54**	35.94	**69.99**	**40.81**	**32.93**	**55.12**	**49.28**
	$KITTI_1$	Baseline	45.92	0.40	21.62	31.55	19.71	29.37	**28.28**	**3.67**	14.54	8.03	33.37	5.79	53.42	23.95	14.23	43.83	22.29	24.46	29.73	20.46	25.13	23.90
		MLDG	65.75	**0.52**	25.59	32.34	8.71	29.71	3.13	2.58	17.40	21.30	15.35	10.51	39.11	17.21	55.21	36.50	35.49	**48.43**	15.27	12.71	29.76	25.27
		SMILE	**92.52**	0.32	27.92	**80.47**	24.57	**61.32**	12.80	0.30	23.85	25.50	7.83	1.76	75.02	30.31	**78.95**	49.10	45.72	36.96	**44.43**	22.63	**43.32**	37.88
		Ours	59.69	0.12	**34.21**	15.91	**33.24**	55.27	24.96	0.08	**78.55**	**32.76**	**77.19**	**17.81**	**88.00**	**37.10**	48.30	**62.57**	**68.11**	21.55	22.86	**36.26**	42.64	**40.96**
	$KITTI_2$	Baseline	16.77	0.28	2.25	39.93	7.95	22.43	**61.30**	**8.32**	42.82	23.41	14.35	**0.99**	38.18	20.47	44.71	35.00	40.48	21.03	20.40	12.54	28.45	24.27
		MLDG	12.29	2.39	1.19	23.54	11.94	22.94	36.96	5.05	73.43	10.97	19.33	0.41	61.90	27.37	57.39	35.11	44.57	**30.05**	29.74	13.58	31.33	26.66
		SMILE	**48.00**	0.23	2.73	**47.63**	**64.24**	**50.98**	48.06	3.54	66.88	36.59	**61.33**	0.15	**74.44**	4.29	**76.32**	37.35	28.16	13.15	21.79	**25.75**	39.79	36.10
		Ours	23.23	**3.29**	**7.81**	0.14	54.90	31.21	34.09	8.09	**93.69**	**39.73**	51.80	0.37	66.14	**40.89**	75.98	**46.54**	**74.04**	25.99	**39.64**	23.39	**42.90**	**37.77**
	$KITTI_3$	Baseline	69.98	**5.70**	22.49	40.30	6.63	3.87	54.55	0.06	28.15	25.90	54.02	0.38	16.69	33.08	24.32	25.35	18.70	**47.13**	18.52	21.69	27.27	26.10
		MLDG	70.29	3.92	1.10	9.10	**27.20**	**4.31**	50.79	0.10	48.12	30.05	28.09	**0.49**	20.51	25.59	41.72	27.76	41.13	22.62	23.74	16.00	27.51	25.09
		SMILE	91.88	2.73	**36.98**	**47.20**	19.18	3.72	52.70	0.02	**86.44**	**33.74**	43.80	0.08	46.27	**58.49**	59.01	41.46	9.92	21.56	24.81	**32.73**	36.60	35.79
		Ours	**92.90**	0.43	29.36	31.16	26.25	0.63	**66.55**	**5.00**	77.93	30.22	**69.06**	0.28	**49.74**	58.00	**83.86**	**60.51**	**66.51**	43.39	**38.38**	27.94	**47.89**	**43.69**
																				Avg	Baseline	16.98	27.78	25.02
																					MLDG	15.72	28.99	25.65
																					SMILE	25.69	39.67	36.05
																					Ours	**30.13**	**47.14**	**42.93**

Table 7: On-the-fly category discovery results on SemanticKITTI within *scenario-B*. The values highlighted in blue are the unobserved-unknown classes. The best value is in **bold**.

	Method	car	bi.cle	mt.cle	truck	oth-v.	pers.	bi.clst	mt.clst	road	park.	sidew.	oth-g	build.	fence	veget.	trunk	terra.	pole	traff.	mIoU Unknown	mIoU Known	mIoU All
Strict	Baseline	71.69	0.46	9.67	**59.40**	12.72	32.67	16.54	0.07	36.76	4.07	38.81	**0.60**	13.93	26.71	47.38	27.07	18.80	15.81	19.20	14.63	24.89	23.81
	MLDG	80.61	0.38	7.53	20.19	**18.53**	41.92	33.07	**1.95**	24.75	17.42	33.44	0.46	27.44	39.63	45.59	20.83	23.16	28.93	22.40	**25.80**	25.68	25.70
	SMILE	0.15	1.57	8.42	35.66	0.70	**49.76**	25.51	0.60	84.24	0.87	44.81	0.46	70.42	**56.87**	48.61	35.91	22.60	46.78	29.71	13.10	31.61	29.67
	Ours	**89.95**	**3.94**	**13.00**	53.69	7.25	41.16	**36.22**	0.89	**88.38**	**34.82**	**70.85**	0.22	**85.38**	38.60	**80.30**	**62.40**	**64.01**	**51.83**	**33.71**	21.73	**47.83**	**45.08**
Greedy	Baseline	73.06	0.46	9.84	**69.51**	26.56	39.58	**40.73**	0.07	36.76	4.07	38.81	**0.60**	13.94	26.73	47.38	27.08	18.80	15.81	19.21	33.64	25.98	26.79
	MLDG	81.76	0.38	1.23	21.14	46.24	42.45	35.27	**2.25**	24.75	17.42	33.45	0.46	27.44	39.63	45.59	20.83	23.16	28.96	22.41	40.75	25.49	27.10
	SMILE	0.21	1.57	8.56	48.16	**65.30**	**52.18**	30.04	0.66	84.24	0.87	44.81	0.46	70.53	**56.89**	49.00	36.13	22.60	46.80	29.74	**47.67**	32.55	34.14
	Ours	**92.87**	**3.99**	**13.87**	66.52	45.21	42.85	39.02	0.93	**88.38**	**34.82**	**70.86**	0.22	**85.61**	38.74	**80.33**	**62.61**	**64.01**	**51.85**	**33.90**	42.12	**48.96**	**48.24**

Table 8: On-the-fly category discovery results on SemanticKITTI within *scenario-C*. The values highlighted in gray and blue are the observed-unknown and unobserved-unknown classes, respectively. The best value is in **bold**.

	Method	car	bi.cle	mt.cle	truck	oth-v.	pers.	bi.clst	mt.clst	road	park.	sidew.	oth-g	build.	fence	veget.	trunk	terra.	pole	traff.	mIoU Unknown	mIoU Known	mIoU All
Strict	Baseline	77.18	5.13	33.99	**70.30**	**16.82**	43.95	15.60	0.11	41.03	2.90	35.77	0.70	71.09	10.61	20.29	28.73	21.21	21.43	8.22	11.77	30.61	27.63
	MLDG	80.91	0.37	27.89	61.06	11.07	**52.57**	19.99	1.04	52.78	4.35	25.09	0.32	46.41	14.43	25.57	**56.32**	31.96	42.48	21.07	11.80	33.77	30.30
	SMILE	90.84	**5.44**	**35.84**	58.40	10.08	50.83	22.35	0.72	72.17	0.81	**51.82**	0.01	73.25	1.73	67.68	41.00	**63.03**	25.44	**29.62**	11.08	41.74	36.90
	Ours	**91.44**	0.29	34.06	37.16	15.65	51.82	**28.94**	**2.15**	**85.94**	**5.39**	46.74	**2.61**	**86.97**	**51.90**	**74.59**	55.58	60.03	**52.08**	23.20	**16.66**	**47.29**	**42.45**
Greedy	Baseline	78.08	5.17	34.28	73.06	21.40	46.47	15.98	0.11	41.57	20.03	35.95	0.72	71.23	10.61	20.29	28.73	21.28	21.43	8.22	19.14	31.07	29.19
	MLDG	84.19	0.38	32.84	64.72	**55.86**	**54.77**	20.01	1.22	53.21	12.54	25.14	0.32	46.41	14.44	25.57	**56.32**	31.96	42.48	21.15	29.47	34.70	33.87
	SMILE	94.10	**5.84**	**38.63**	**80.64**	49.73	52.16	**33.03**	0.80	75.05	**72.83**	**52.65**	0.01	73.60	41.00	67.73	41.01	**63.20**	25.44	**29.63**	**51.86**	43.89	45.15
	Ours	**94.40**	0.29	37.62	33.14	43.32	53.24	30.68	**2.35**	**88.02**	37.56	47.20	**2.66**	**87.14**	**52.13**	**74.61**	55.65	60.14	**52.11**	23.38	37.19	**47.76**	**46.09**

2 Class-wise Results

As mentioned in Sec. 4.1 of the main paper, we conducted experiments in various scenarios to demonstrate the validity of our proposed method. The class-wise IoU results and mIoU for each scenario are presented in Tabs. 5 to 8.

Analysis for known classes. In the experiments conducted for *scenario-A* on the SemanticPOSS and SemanticKITTI, our method outperforms baselines on known classes. Specifically, when comparing the class-wise results of known classes for each split of the SemanticPOSS with other baselines, our method exhibits the highest performance in a majority of cases (29 out of 39 known classes in POSS_0-POSS_3 in Tab. 5) in Strict-Hungarian. Similarly, for the SemanticKITTI, when comparing the class-wise results of known classes for each split (37 out of 57 known classes in KITTI_0-KTTI_3 in Tab. 6), our method demonstrates the highest performance in most cases in Strict-Hungarian. For the results of *scenario-B* shown in Tab. 7, our method performs better than others for 12 out of 17 known classes in Strict-Hungarian. In *scenario-C* of Tab. 8, where observed-unknown and unobserved-unknown classes coexist, our method exhibits the highest performance for 8 out of 16 known classes in Strict-Hungarian. Thus, our proposed method consistently shows superior performance on known classes across various scenarios when compared to different baselines. Notably, the minor performance difference of known classes between Strict- and Greedy-Hungarian metrics, particularly in *scenario-B* and *C*, suggests that the primary cause of the lower performance on known classes of baselines is not due to confusion between known and unknown classes. Instead, the lower performance of known classes can be attributed to 1) the high sensitivity of hash code to intra-class variation and 2) confusion among known classes. Considering the performance of baseline and MLDG on known classes in Tabs. 5 to 8, along with the qualitative results, it can be interpreted that these methods are vulnerable to intra-class variation. Unlike baseline and MLDG, SMILE benefits from sign-magnitude disentanglement proposed in [2], resulting in a reduction in misclassifications due to intra-class variation. However, it still exhibits confusion among known classes, leading to lower performance compared to the proposed method. In conclusion, the proposed method demonstrates robustness to both sources of problems, intra-class variation and class confusion, as confirmed quantitatively and qualitatively.

Analysis for unknown classes. In various scenarios, the proposed MCL consistently demonstrates high performance in unknown classes. As shown in Tabs. 5 and 6, our method shows meaningful gain over other baselines on unknown classes (both in Strict- and Greedy-Hungarian). For the *scenario-A*, the proposed model particularly shows significant improvement in observed-unknown classes such as *plants*, *car*, *building*, and *bike*. In some splits, however, specific unknown classes exhibit lower performance in Strict-Hungarian than other baselines. As the model gains a better understanding of the semantics for known classes, without a specific solution, it may lead to potential confusion when assessing unknown classes since the unknown classes are cognized based on the knowledge acquired from the known classes. Considering this performance trade-off between known and unknown classes, the most challenging aspect of OCDSS

is the ability to cognize unknown classes without compromising the performance of known classes. In the case of SMILE, it enhances recognition ability for known classes in exchange for cognition ability for unknown classes. Although we can observe slightly lower performance compared to SMILE in KITTI_3, the proposed method demonstrates its effectiveness by achieving a remarkable 14% higher performance for known classes in Strict-Hungarian. Considering the aforementioned trade-off between known and unknown classes, this suggests that the proposed method remains effective or even superior. In *scenario-B*, as shown in Tab. 7, MLDG exhibits slightly higher performance on unknown classes compared to our method. Though the meta-learning-based approach in MLDG helps to learn unobserved-unknown classes, it results in inadequate representation learning for known classes. In contrast, our method manages to achieve meaningful gains not only in known classes but also in unknown classes. Upon the *scenario-C*, our method performs well on both observed-unknown classes (*other-vehicle* and *parking*) and unobserved-unknown class (*bicyclist*) which is only *cognizable* by OCDSS. Our method demonstrates superior performance compared to other methods in *scenario-C*, according to the Strict-Hungarian. However, according to the Greedy-Hungarian, SMILE performs better in unknown classes. This can be attributed to SMILE's proficiency in clustering semantic classes well in simple situations where known and unknown classes are clearly separated, but it struggles in complicated scenarios where known and unknown classes coexist. For instance, SMILE exhibits lower performance in *parking* under Strict-Hungarian compared to the notably higher performance observed in the Greedy-Hungarian. This observation suggests that SMILE effectively clusters *parking* into a single cluster when known-unknown distinctions between *parking* and *road* are guided by the ground truth. However, when *parking* coexists with *road*, SMILE struggles to differentiate *parking* from *road* or *sidewalk* accurately. Considering that ground truth-based separation between known and unknown classes is impossible in real situations, and confusing classes coexist, our method demonstrates practicality and effectiveness by achieving high performance according to the Strict-Hungarian.

3 Implementation Details

3.1 Proposed Method

We use MinkUNet34 [1] as the feature extractor $\mathcal{E}$. For the projection head ϕ, we use three multi-layer perceptrons (MLP) where the channel dimension is (96,256,256,32) followed by a single MLP layer that maps the features into projection space with the dimension $L = 12$. We voxelize the point cloud using a voxel size of 0.05m. We use SGD optimizer [6] with learning rate $lr = 0.01$, momentum 0.9, and weight decay $1e-4$ with the batch size of 4. During training, we use a fixed mixing ratio $r = 0.1$ in the mixing module. For the temperature parameters, $\tau_{cos} = 0.07$ and $\tau_{hash} = 0.14$ are used. We use $\lambda = 0.1$ for SemanticKITTI and $\lambda = 0.05$ for SemanticPOSS in *scenario-A*. In addition, we use $\lambda = 0.01$ in *scenario-B* and *scenario-C*. For data augmentation, we augment

the data with random downsampling of the points, random scaling, and random rotation along the z-axis during training. Additionally, we do not use the intensity value of the points.

3.2 Baselines

For a fair comparison, we use the same backbone architecture, optimizing strategy, and data augmentation for all baselines and the proposed method.

MLDG [5]. MLDG is a meta-learning method for domain generalization. As the proposed OCDSS tends more towards generalizing to unknown classes from known classes, we simulate the presence of unknown classes during training and apply the MLDG method to the baseline. Specifically, at each training iteration, we divide the existing known classes in half, with one half used as known classes appearing in the meta-train and the remaining used as unknown classes appearing in the meta-test. MLDG involves hyperparameters α and β, where α represents the meta-train step size and β is a weighting parameter that balances meta-train and meta-test. Following the original paper, we experiment with values such as $\alpha = 0.01$, the same as the learning rate, and $\beta = 1$, achieving the best performance.

SMILE [2]. SMILE proposes sign-magnitude disentanglement to reduce the sensitivity of hash codes to intra-class variance and capture the class-level semantics better. For SMILE, we use the same architecture for the feature extractor as the proposed method (*i.e.* MinkUNet34) while the projection head ϕ is modified by following the original paper. Instead of the final single MLP layer in ϕ, two separate MLP layers are employed for sign and magnitude predictions. Then, the representation is obtained by the element-wise multiplication of sign and magnitude. SMILE involves hyperparameter α that weights the magnitude regularization loss for sign predictions. When the value $\alpha = 3$, as used in the original paper, is employed, the magnitude regularization loss dominates the training procedure, causing instability in the training process and resulting in lower performance compared to the baseline. Therefore, we search appropriate values for α, and use $\alpha = 0.1$, which achieves the best performance.

3.3 Computational Cost Analysis

In Tab. 9, we report the details regarding computational cost and the number of parameters for each method. Our method additionally uses only negligible parameters compared to other baselines, resulting in a nearly identical total number of parameters. However, during the training process, the time taken per iteration (s/it) increases slightly compared to the baseline. The MLDG method, which performs meta-learning during training, requires the most time and memory. During inference, the time and memory usage are the same since the same backbone is used.

Table 9: Computational cost and number of parameters for each method. The cost is computed on SemanticPOSS.

		Training		Inference	
	Parameters	Time	Memory	Time	Memory
Baseline	37.9M	0.66s/it	10.6GB	0.086s/it	2.6GB
MLDG		2.06s/it	25.6GB		
SMILE		0.67s/it	10.8GB		
Ours		0.85s/it	12.1GB		

3.4 Data Split

In the supplementary material, we elaborate on the details of the class selection process in *scenario-A*. To assess the discovery capability for various observed-unknown classes, we make four distinct splits each for SemanticKITTI and SemanticPOSS in *scenario-A*, depending on the choice of the unknown class. Specifically, we consider the diversity in the size and geometry of the unknown classes and use these considerations to select classes for each split. For example, in SemanticPOSS, we distribute massive classes such as *plants*, *building*, and *ground* into separate splits, considering the diversity in class size. Similarly, in SemanticKITTI, we distribute classes like *vegetation*, *road*, *sidewalk*, and *building* into distinct splits. Notably, as OCDSS generalizes from known to unknown classes, selecting massive classes altogether as unknown during training may hinder meaningful learning. Therefore, we prevent this by distributing massive classes across different splits. Additionally, we structured the unknown classes in each split to have a diversity of geometries. Therefore, as examples, POSS_0 and KITTI_0 are composed of {*trashcan*, *traffic sign*, *bike*, *plants*} and {*bicycle*, *other-vehicle*, *trunk*, *terrain*, *vegetation*}, respectively, ensuring that the unknown classes have diverse sizes and geometries.

3.5 Evaluation Metrics

We adopt Greedy-Hungarian [3] and Strict-Hungarian [7] evaluation protocols by following [2]. During testing, the class descriptor from the hash code directly forms clusters, which are then matched with ground truth using the Hungarian matching algorithm [4]. Clusters that are not matched are regarded as misclassified. Then, the intersection over union (IoU) for each class and mean values (mIoU) are calculated for evaluating semantic segmentation performance. In the **Greedy-Hungarian**, we first divide the points into known and unknown classes based on the ground truth before evaluation, and then perform Hungarian matching separately within each group. Consequently, in the Greedy-Hungarian criterion, there may exist cases where a single class descriptor (*i.e.* hash code) is simultaneously assigned to both known and unknown classes. Therefore, Greedy-Hungarian evaluates the ability to form semantically meaningful clusters within the known and unknown groups *separately*, excluding the ability to distinguish between known and unknown classes. On the other hand, the **Strict-Hungarian** conducts Hungarian matching for the entire set of classes without distinguishing

between known and unknown classes. Accordingly, a single class descriptor (*i.e.* hash code) is assigned to only one class among the entire set. Strict-Hungarian evaluates both the ability to form semantically meaningful clusters and the ability to distinguish between known and unknown classes simultaneously. This better reflects real-world scenarios where known and unknown classes are not distinguished and appear together, making it a more practical criterion compared to Greedy-Hungarian.

4 Additional Qualitative Results

We provide additional qualitative results for three scenarios in Figs. 1 to 3. Figures 1 and 2 respectively show the results on SemanticPOSS and SemanticKITTI within *scenario-A*, and Fig. 3 show the results in *scenario-B* and *scenario-C*. The results are visualized using the Strict-Hungarian evaluation protocol. In general, both the baseline and MLDG show high misclassification rates across all scenarios due to their vulnerability to intra-class variance. While SMILE reduces misclassifications compared to the baseline and MLDG, it still reveals some confusion between classes and struggles to classify and segment unknown classes. On the other hand, our proposed method demonstrates successful segmentation for both known and unknown classes compared to the baselines.

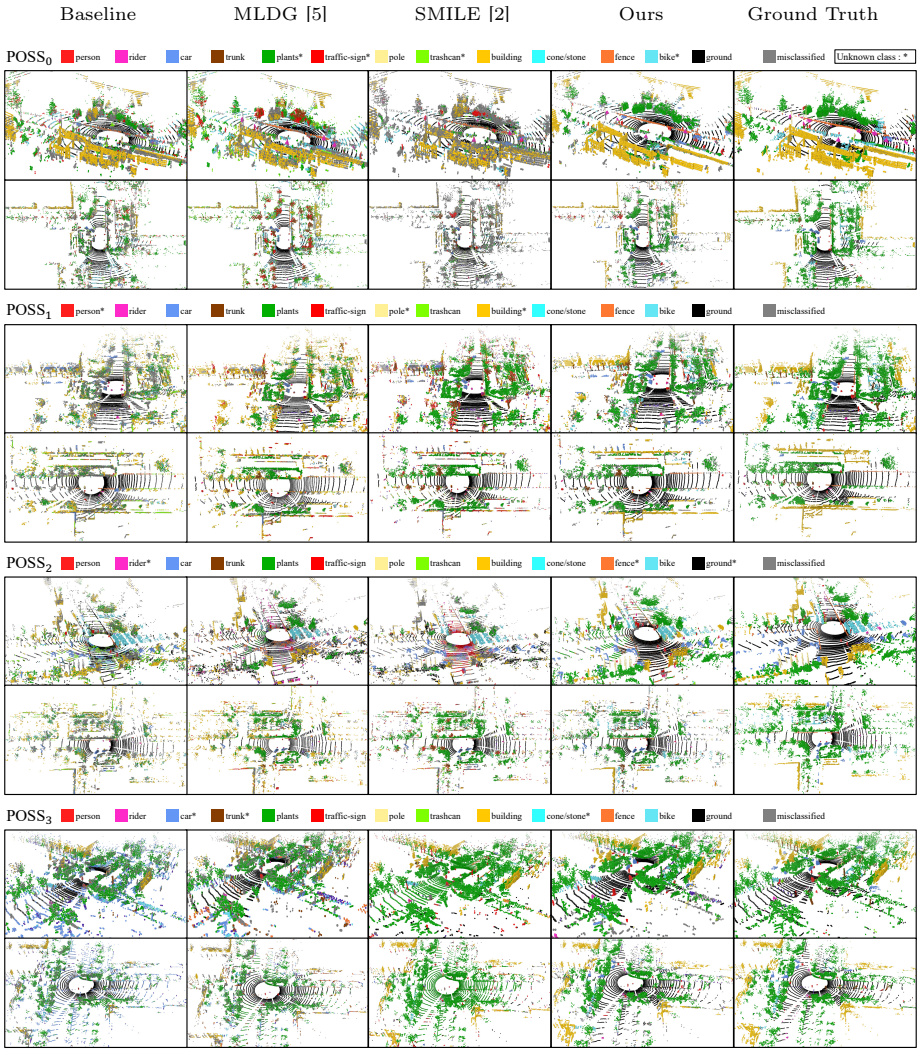

Fig. 1: Qualitative results on SemanticPOSS within *scenario-A*. Best viewed when zoomed in with colors.

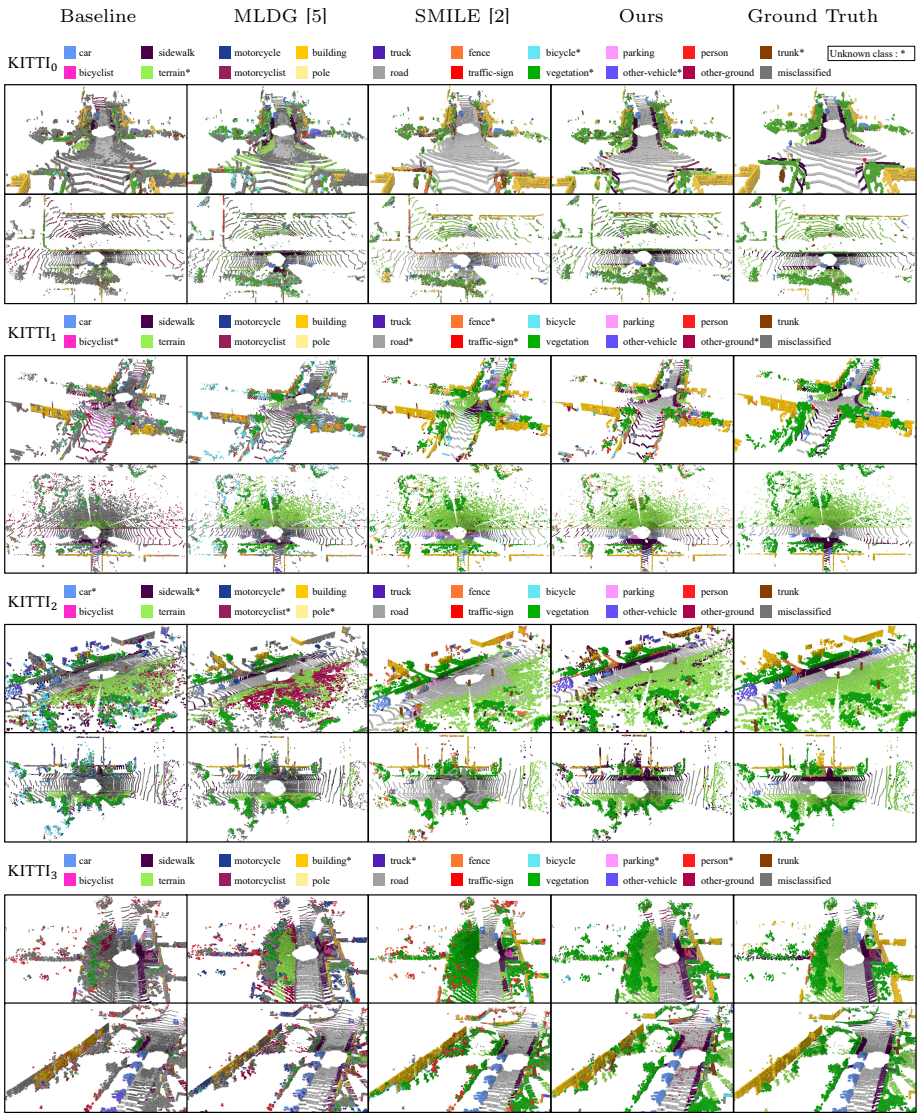

Fig. 2: Qualitative results on SemanticKITTI within *scenario-A*. Best viewed when zoomed in with colors.

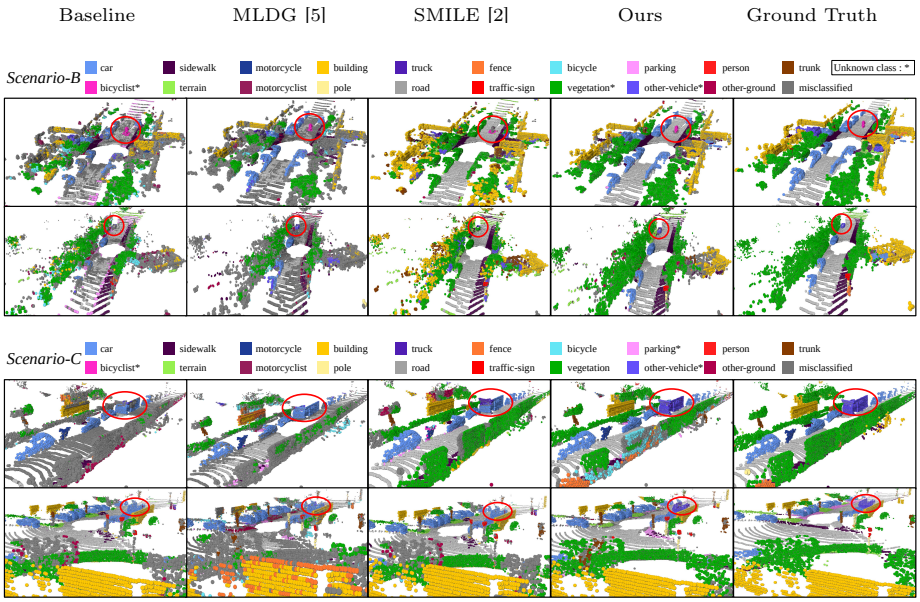

Fig. 3: Qualitative results on SemanticKITTI within *scenario-B* and *scenario-C*. The red circle refers to the part of unknown class. Best viewed when zoomed in with colors.

References

1. Choy, C., Gwak, J., Savarese, S.: 4d spatio-temporal convnets: Minkowski convolutional neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3075–3084 (2019)
2. Du, R., Chang, D., Liang, K., Hospedales, T., Song, Y.Z., Ma, Z.: On-the-fly category discovery. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11691–11700 (2023)
3. Fini, E., Sangineto, E., Lathuiliere, S., Zhong, Z., Nabi, M., Ricci, E.: A unified objective for novel class discovery. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9284–9292 (2021)
4. Kuhn, H.W.: The hungarian method for the assignment problem. Naval research logistics quarterly **2**(1-2), 83–97 (1955)
5. Li, D., Yang, Y., Song, Y.Z., Hospedales, T.: Learning to generalize: Meta-learning for domain generalization. In: Proceedings of the AAAI conference on artificial intelligence. vol. 32 (2018)
6. Ruder, S.: An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747 (2016)
7. Vaze, S., Han, K., Vedaldi, A., Zisserman, A.: Generalized category discovery. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7492–7501 (2022)
8. Verma, V., Lamb, A., Beckham, C., Najafi, A., Mitliagkas, I., Lopez-Paz, D., Bengio, Y.: Manifold mixup: Better representations by interpolating hidden states. In: International conference on machine learning. pp. 6438–6447. PMLR (2019)
9. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond empirical risk minimization. arXiv preprint arXiv:1710.09412 (2017)