Simple Multi-dataset Detection — Supplementary Materials

A. Dataset details

Table 1 lists the datasets we used in our experiments. We use the Robust Vision Challenge[1] official release of each dataset. Specifically, we use the standard 2017 train/ validation split for COCO [7], the Challenge-2019 release of OpenImages [6], and the default version of Objects365 [10] and Mapillary [8]. For ScanNet [3], as there is no standard train/ validation split, we use the first 80% scenes (sorted by scene ID) as training and the last 20% scene as validation. For KITTI [5], we used the RVC challenge version that has instance-segmentation version, which contains 200 images. For WildDash [12], we use the public version for evaluation, and report standard mAP performance. We don't consider the negative label metric in the official website. For CrowdHuman [11], we use the visible bounding box annotation, and report mAP instead of the missing rate as the official metric. We use the official train/ validation split and the official evaluation metrics for VIPER [9], Cityscapes [2], and Pascal VOC [4].

[1] http://www.robustvision.net

Dataset name	Domain	# Cat.	# Img.
Train & Validation			
COCO	Internet images	80	118k
Objects365	Internet images	365	600k
OpenImages	Internet images	500	1.8M
Mapillary	Traffic	38	18k
Test			
ScanNet	Indoor	20	25k
VIPER	Virtual	10	13k
Cityscapes	Traffic	8	12k
WildDash	Traffic	13	4k
KITTI	Traffic	8	200
Pascal VOC	Internet images	20	16k
CrowdHuman	Internet images	1	15k

Table 1. Datasets we used in training and testing. Top: datasets we used in training and validation, which are from the Robust Vision Challenge. Bottom: datasets we used for zero-shot cross-dataset testing.

B. Computation of label space learning algorithm and pruning

The size of our optimization problem scales linearly in the number of potential merges $|\mathbb{T}|$, which can grow exponentially in the number of datasets. To counteract this exponential growth, we only consider sets of classes

$$\mathbb{T}' = \left\{ \boldsymbol{t} \in \mathbb{T} \middle| \frac{c_{\boldsymbol{t}}}{|\boldsymbol{t}| - 1} \leq \tau \right\}.$$

For an aggressive enough threshold τ, the number of potential merges $|\mathcal{T}'|$ remains manageable. We greedily grow $\mathcal{T}'$ by first enumerating all feasible two-class merges ($|\boldsymbol{t}| = 2$), then three-class merges, and so on. The detailed algorithm diagram is shown in Algorithm 1. The runtime of this greedy algorithm is $O(|\mathcal{T}'| \max_i |\hat{L}^i|)$. In practice, the cost computation took a few seconds for the distortion loss function and about 10 minutes for the AP loss (due to the need to repeatedly recompute AP). The integer programming solver finds the optimal solution within one second in both cases.

C. Adding new datasets to a label space

While we tend to keep the training domains and label space large and comprehensive, it is inevitable in practice that more fine-grained labels or specific testing domains are needed. Given a learned a unified label space on an existing set of training datasets, we use a simple label space expansion algorithm to allow adding more datasets and labels after the unified detector is trained.

Similar to our unified label space learning algorithm, we run the unified detector on the new training data. We evaluate the AP between each class in the new dataset annotation and each class in the unified label space. We merge the new class into the existing class that gives the lowest merge cost (Section. 4.2). In our experiments, add Mapillary dataset [8] to our label space we using the AP loss. If the cost is lower than a threshold (AP change < 5 AP in our implementation). Otherwise, we append the new class to the unified label space as a single class.

D. Discussion on label hierarchy

Different datasets may contain different label granularities for the same concept, and there exists label hierar-

	COCO	CityScapes	Mapillary	VIPER	ScanNet	OpenImages	KITTI	WildDash
COCO	**35.6**	19.6	3.2	8.5	5.2	7.2	15.7	8.4
CityScapes	0.0	21.5	0.8	2.3	0.0	0.0	13.0	2.4
Mapillary	0.6	11.7	**10.6**	9.0	1.2	0.0	13.4	5.4
VIPER	0.1	2.8	1.1	**17.8**	0.0	0.0	6.5	1.4
ScanNet	0.4	0.0	0.0	0.0	**35.6**	0.0	0.0	0.0
OpenImages	12.9	9.5	1.1	3.5	1.7	**52.8**	7.2	4.9
Unified (ours)	24.0	**28.3**	8.1	16.5	28.7	41.8	**16.9**	**11.3**

Table 2. Instance segmentation performance on six training datasets and two new datasets (KITTI and WildDash). We show mask mAP on the validation set of each dataset.

Algorithm 1: Learning a unified label space

Input : $\{\mathbf{b}_i, \mathbf{l}_i\}_{i=1}^N$: ground truth bounding boxes and labels for each of the N training datasets
$\{\{\tilde{\mathbf{b}}_i^{(j)}, \tilde{\mathbf{l}}_i^{(j)}\}_{j=1}^N\}_{i=1}^N$: predicted bounding boxes with predicted classes in all datasets for each training dataset
λ, τ: hyper-parameters for algorithm

Output: L: unified label space
$\mathcal{T}$: the transformation from each individual label space to the unified label space

// Compute potential merges and merge cost
$\hat{L} = \bigcup_i \hat{L}_i$ // Short-hand used to simplify notation
$\mathbb{T}_1 \leftarrow \{(l) | l \in \hat{L}\}$ // Set of single labels
Compute $c_{\boldsymbol{t}}$ for all single labels $\boldsymbol{t} \in \mathbb{T}$. // 0 for most metrics
for $n = 2 \dots N$ **do**
  $\mathbb{T}_n \leftarrow \{\}$
  for $\boldsymbol{t} \in \mathbb{T}_{n-1}$ **do**
    for $l \in \hat{L}$ **do**
      if *l and all labels in $\boldsymbol{t}$ are from different datasets* **then**
        compute $c_{\boldsymbol{t} \cup \{l\}}$.
        if $\frac{c_{\boldsymbol{t} \cup \{l\}}}{n-1} \leq \tau$ **then**
          Add $\boldsymbol{t} \cup \{l\}$ to $\mathbb{T}_n$.
        end
      end
    end
  end
end
$\mathbb{T} \leftarrow \bigcup_{n=1}^N \mathbb{T}_n$
// Solve the ILP.
$\boldsymbol{x} \leftarrow$ ILP_solver$(c, \mathbb{T}, \lambda)$ // Solve equation (8).
Compute $L, \mathcal{T}$ from $\boldsymbol{x}$
Return: $L, \mathcal{T}$

chies inter or intra datasets. For example, Objects365 [10] does not have a "bird" category, but has more fine-grained bird species like parrot, pigeon, and swan, while most other datasets only annotate "bird". Our label space optimization algorithm automatically handles the hierarchical label space issue: the fine-grained birds in Objects365 will not merge with COCO birds because this merge introduces many false-positives for the fine-grained birds in Objects365 and yields a large cost. Our unified label space will contain both the general "bird" class and each fine-grained class. The model trained on the unified label space is expected to predict both the coarse "bird" label and the fine-grained label in testing.

E. Instance segmentation

We further evaluate our label space learning algorithm and unified training framework on instance segmentation. We follow the Robust vision challenge set up to use 8 datasets: COCO, OpenImages, Mapillary, ScanNet, VIPER, CityScapes, WildDash and KITTI (the same as Table 1, except OpenImages segmentation set has 300 instead of 500 classes.). Again, we leave WildDash and KITTI as testing only as they are small and similar to CityScapes and Mapillary. We run our label space learning algorithm (Section. 4) on the remaining six datasets, resulting a unified label space of 358 classes. We use CascadeRCNN [1] with a standard mask head as the detector, and train a $2\times$ schedule with ResNet50. The dataset-specific models are trained with $1\times$ or $2\times$ schedule depending on their size.

Table. 2 compares the unified detector to dataset specific models. As expected, no single dataset-specific model performs well on all test domains. Our unified model performs consistently good on all training datasets. More importantly, it generalizes the best to the new test datasets (KITTI and WildDash) than any single dataset model.

References

[1] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: High quality object detection and instance segmentation. *TPAMI*,

2019. 2

[2] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 1

[3] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, 2017. 1

[4] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The PASCAL visual object classes (VOC) challenge. *IJCV*, 2010. 1

[5] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012. 1

[6] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, Tom Duerig, and Vittorio Ferrari. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *IJCV*, 2020. 1

[7] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 1

[8] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulò, and Peter Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *ICCV*, 2017. 1

[9] Stephan R Richter, Zeeshan Hayder, and Vladlen Koltun. Playing for benchmarks. In *ICCV*, 2017. 1

[10] Shuai Shao, Zeming Li, Tianyuan Zhang, Chao Peng, Gang Yu, Xiangyu Zhang, Jing Li, and Jian Sun. Objects365: A large-scale, high-quality dataset for object detection. In *ICCV*, 2019. 1, 2

[11] Shuai Shao, Zijian Zhao, Boxun Li, Tete Xiao, Gang Yu, Xiangyu Zhang, and Jian Sun. Crowdhuman: A benchmark for detecting human in a crowd. *arXiv:1805.00123*, 2018. 1

[12] Oliver Zendel, Katrin Honauer, Markus Murschitz, Daniel Steininger, and Gustavo Fernandez Dominguez. Wilddash-creating hazard-aware benchmarks. In *ECCV*, 2018. 1