Scaling Object Detection by Transferring Classification Weights

Jason Kuen[1] Federico Perazzi[2] Zhe Lin[2] Jianming Zhang[2] Yap-Peng Tan[1]
[1]Nanyang Technological University, Singapore [2]Adobe Research

Abstract

Large scale object detection datasets are constantly increasing their size in terms of the number of classes and annotations count. Yet, the number of object-level categories annotated in detection datasets is an order of magnitude smaller than image-level classification labels. State-of-the art object detection models are trained in a supervised fashion and this limits the number of object classes they can detect. In this paper, we propose a novel weight transfer network (WTN) to effectively and efficiently transfer knowledge from classification network's weights to detection network's weights to allow detection of novel classes without box supervision. We first introduce input and feature normalization schemes to curb the under-fitting during training of a vanilla WTN. We then propose autoencoder-WTN (AE-WTN) which uses reconstruction loss to preserve classification network's information over all classes in the target latent space to ensure generalization to novel classes. Compared to vanilla WTN, AE-WTN obtains absolute performance gains of 6% on two Open Images evaluation sets with 500 seen and 57 novel classes respectively, and 25% on a Visual Genome evaluation set with 200 novel classes.

1. Introduction

State-of-the-art object detectors [12, 34] are typically trained with a large number of bounding box annotations. Large-scale datasets such as COCO [26], Pascal VOC [7] and OpenImages [22] provide a substantial amount of bounding boxes, but the number of annotated object categories is often very limited. The reason is that scaling the number of bounding boxes can be semi-automated, *e.g.* [22], while increasing the number of classes requires significant human labor. On the other hand, image-level labels such as those available in classification datasets are much easier to collect as they do not require costly bounding box annotations. As a consequence, several works investigated the training of object detectors in a weakly-supervised regime, using only image-level labels. These methods leverage a variety of classes available in classification datasets or image tags found in social networks [29] but neglect the spatial information available in object detection datasets.

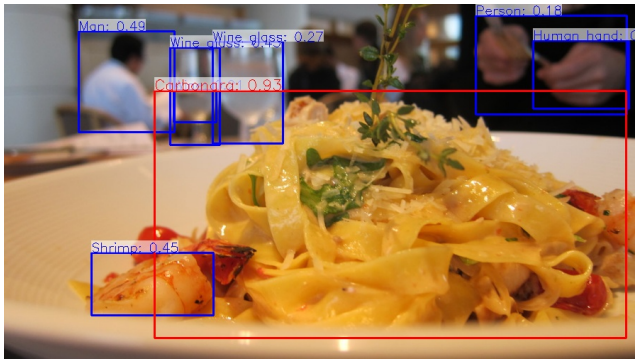

Figure 1. Our proposed detector has no access to box-level training annotations for the object class represented by the red box, "*Carbonara*". It learns to detect novel object classes by transferring weight knowledge from large-scale pre-trained image classification network.

In contrast, partial supervised methods [16] employ both types of annotations. While existing methods [24, 33, 39, 40] that transfer knowledge from a classification network to a detection network with partial supervision achieve higher accuracy than weakly-supervised methods [3, 4, 24], they incur a significant computational cost during training and testing. The overhead comes either from joint training of the two networks [24, 33], or from performing forward passes of the classification network during testing [39, 40]. Furthermore, joint-training methods often require storage-intensive, large-scale classification datasets to be present while training the detection network.

To overcome these limitations, we propose a novel approach to transfer discriminative semantic knowledge from classification to detection with a non-linear weight-transfer network (WTN) [16]. Given a set of common classes annotated for both tasks, we learn a function, the weight-transfer network, that maps weights at the fully-connected layer of the classification network to those of the object detection network. Once trained, WTN is used to extend the number of categories recognized by the object detector via transferring weights of unseen classes from the classification network. This strategy is advantageous because it only adds little computational and memory overheads to training and

no burden to inference at all.

Compared to the vanilla weight-transfer network [16], we introduce two key components to our model. First, we insert normalization layers to account for the different amplitude of the classification weights. Secondly, we replace the multilayer perceptron with an autoencoder. The latent space of the autoencoder corresponds to the classification weights of the object detector and therefore is trained with object-level supervision. The reconstruction loss between the input and output of the autoencoder is essential to retain semantic information of all the classes while the detection network's classification loss facilitates the learning of a discriminative embedding of the class weights.

Extensive experimentation on Open Images [22] and Visual Genome [20] datasets demonstrates that the proposed method significantly outperforms existing partially-supervised detection approaches on challenging detection tasks involving novel object classes. Moreover, due to the auxiliary regularization effect brought by the reconstruction loss of autoencoder WTN, our proposed method even recovers the performance loss of existing WTN on seen classes.

Contributions. The contributions of this work are threefold: *i)* we address the under-fitting issue of WTN by introducing input and feature normalization schemes. The resulting model WTN$^+$achieves improved detection performance over the vanilla WTN; *ii)* we propose our main model, autoencoder WTN, that better preserve semantic knowledge of all object classes, while learning to generate discriminative classification weights for the detection network; *iii)* we verify the effectiveness of our method with extensive evaluations using large-scale datasets with millions of images and several hundreds of object classes.

2. Related Works

Over the years, several convolutional network-based object detection frameworks and architectures have been proposed: R-CNN [10], Fast R-CNN [9], Faster R-CNN [34], R-FCN [5], SSD [27], YOLO [32, 33], FPN [25]. They can be roughly categorized into single-shot detectors [5, 27, 32, 33] which predict detection boxes from feature maps directly, and two-shot detectors [9, 10, 34] which first generate object proposals and then perform spatial extraction of feature maps based on the proposals for further predictions. These approaches have improved object detection from an algorithmic perspective and in a fully supervised setting. In this work, we adopt Faster R-CNN [34] because its box-level *classification head* learns just a single set of classification weights, resembling image-level classification (source task) networks. This allows a smoother knowledge transfer from classification to detection, compared to using single-shot detection networks which learn multiple sets of classification weights for different anchor boxes.

Object-levels annotations are time-consuming and tedious to collect, especially when the number of classes is large. With a large number of classes, it is very challenging to obtain accurate and complete annotations due to complex overlapping meanings of classes. Thus, several approaches attempt to scale up the number of object classes handled by object detectors using image-level annotations. Transferring knowledge from image classification to object detection is an active research area tackling the lack of bounding box annotations of the target datasets and/or object classes. These knowledge transfer-based methods for scaling up object detection can be divided into two categories: weakly-supervised and partially-supervised approaches.

Weakly-supervised methods typically rely only on an image-level classification dataset and leverage class agnostic box proposals or prior object knowledge to build object detectors. For example, Uijlings *et al*. [43] perform multiple instances learning with knowledge transfer (source dataset with bounding boxes) to produce boxes for the target training dataset. In [41], a weakly-supervised object detector is trained on a weakly-labeled web dataset to generate pseudo ground-truths for the target detection task. [37] combines region-level semantic similarity and common-sense information learned from some external knowledge bases to train the detector with just image-level labels.

More closely related to our work are weight adaptation methods [15, 39, 40] that fine-tune classification networks and learn detection-specific bias vectors to adapt the networks for detection. These adaptation-based methods assume the classification power of the network is well-preserved (e.g., using R-CNN [10]) when transferred to the detection task. This restricts them from being effectively applied to recent detection methods (e.g., Faster R-CNN [34], feature pyramid network [25]) that significantly modify the backbone network structure. Whereas, our method is not restricted by such constraints.

In general, classification weight-based knowledge transfer [16] can be applied to any recent detection frameworks [27, 33, 34]. On the other hand, partially supervised approaches employ weak labels, i.e. image-level annotations, as well as bounding box-level annotations. For example, YOLO-9000 [33] extend the detector's class coverage by concurrently training on bounding box-level data and image-level data, such that the image-level data contribute only to classification loss. By decoupling the detection network into two branches (positive-sensitive & semantic-focused), R-FCN-3K [37] is able to scale detection up to 3000 classes despite being trained on limited bounding box annotations for several object classes. In contrast to these, we focus on large-scale object detection without having access to additional data (classification) sources during the training. A well-trained image classification network possesses sufficiently rich semantic knowledge about the large-

scale dataset's categories and the information is compressed in weights of its classification layers. We argue that such weights can effectively be exploited to help build an object detector handling a large number of categories.

3. Weight Transfer Network

Preliminaries. We consider the setting of a classification network CLN that handles object classes C, and a detection network DEN that handles object classes D. The number of categories handled by CLN is much greater than the number of categories handled by DEN, i.e. $|C| >> |D|$. The goal of our approach is to expand the number of categories handled by DEN through partial supervision, where we transfer weight knowledge from CLN (source task) to DEN (target task). We make use of the final fully-connected (FC) layer weights of the CLN that has been pre-trained on a large scale image classification dataset. The final FC layer weights can be seen as a form of *semantic embeddings* comprising rich knowledge about the object categories and the complex class relationships. Furthermore, pre-trained large-scale image classification networks are very accessible and many are shared publicly.

Classification knowledge from CLN is transferred to DEN using a weight transfer network (WTN) through the object categories shared (S) between the two tasks: $S = C \cap D$. WTN is a neural network that works as a class-generic function $T()$ used to transform per-class classification weight vectors $W_C = [w_C^1, w_C^2, ..., w_C^{|C|}]$ from CLN to DEN's classification weights $W_D = [w_D^1, w_D^2, ..., w_D^{|D|}]$ as follows: $W_D = T(W_C)$.

WTN is trained jointly with DEN on detection dataset with classes D. The gradients of WTN's network parameters come from DEN's box-level classification loss ℓ_{cls}. Before training WTN and DEN, we 'freeze' W_C (taken from pre-trained CLN). While S rely on WTN, for the DEN's categories which are not part of S (i.e., $D \setminus S$), we train their weights as in conventional detection network. To obtain DEN's classification score predictions, we simply perform matrix multiplication between DEN's box-level visual features and WTN's predicted weights, just like how it works for conventional classification weights. Conventionally, WTN is based on a two-layer multi-layer perceptron (MLP) architecture.

Due to its class-genericness, WTN is able to carry out effective *inductive learning* [6]. In other words, despite that only classes S are seen by WTN and DEN during training, during testing WTN (and the DEN model that incorporates WTN) can work reasonably well with classes N of CLN that are not shared with DEN, i.e. $N = C \setminus S$.

Normalizations. Large-scale classification datasets have an unbalance class distribution, which has strong implications in how the classification weights of CLN are trained. E.g., in one large-scale CLN, we discover that the 'highest-norm' class has a weight vector *norm* that is 28 times that of the 'lowest-norm' class. Besides, a *class-generic non-linear* WTN naturally cannot adapt and learn as well as (conventional) *class-specific linear* classification weights, for loss minimization. These pose challenges to the training and optimization of WTN. Empirically, we found that training a detection network (DEN) with existing WTN methods deteriorates the performance on D classes, compared to a conventional DEN trained on the same labels but without WTN.

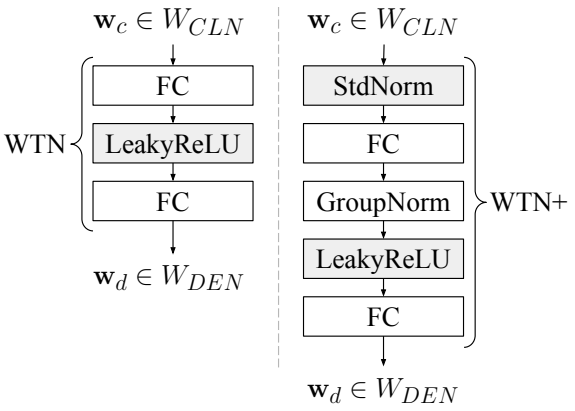

Figure 2. Comparison between network architectures of WTN and WTN$^+$. The white rectangles correspond to layers with learnable parameters.

Thus, drawing from the recent findings in activations normalization techniques [17, 44], we introduce a new variant of WTN, WTN$^+$ that improves performance on D classes and it is easier to optimize. The model architectural differences between WTN and WTN$^+$are illustrated in Fig. 2. Standard normalization is applied to the input weights W_C to enable different input channels to contribute comparably to the prediction of W_D, in order to curb the overdominance/underdominance of certain categories. Let v_j denote the weights of j-th feature/channel of W_C, we normalize v_j by: $\frac{v_j - \mu(v_j)}{\sigma(v_j)}$, where $\mu(\cdot)$ and $\sigma(\cdot)$ are the *mean* and *standard deviation* functions respectively. Group Normalization [44] layer, known for its strong optimization benefits, is added to normalize intermediate features to encourage good gradient flows for easier network optimization. These small but crucial modifications are the key to training highly effective WTN.

4. Autoencoder Weight Transfer Network

During training, only the shared classes S contribute to the gradients and losses of WTN. The novel object classes N are unknown to and unconsidered by WTN. The lack of knowledge of the entire class population of C limits WTN's capability to effectively model the *good* classification space

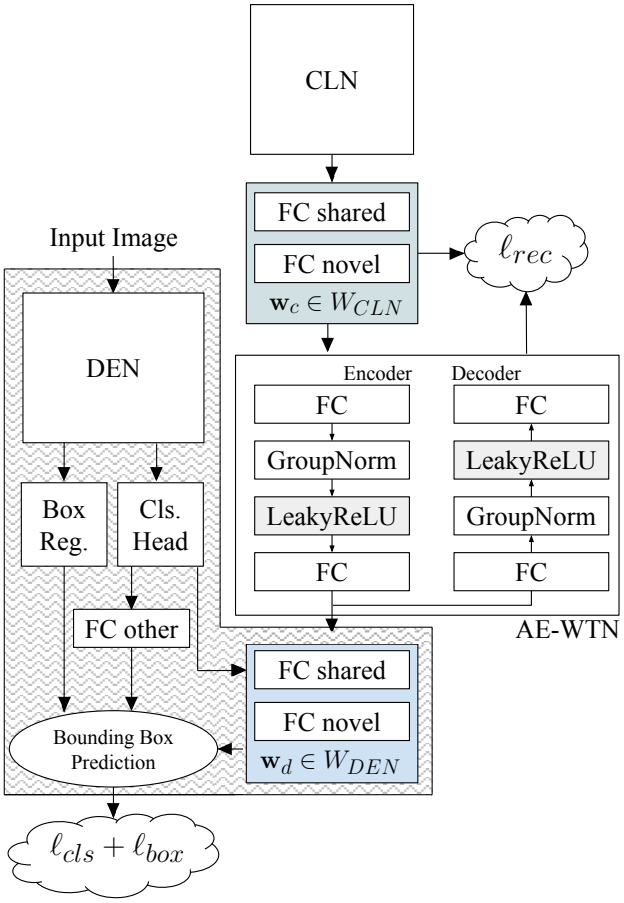

Figure 3. The *train* and *test* phases of object detector (DEN) with an Autoencoder-WTN (AE-WTN). *Train phase*: Before training DEN, we extract CLN's final FC layer's weights W_C, and discard the earlier layers. Trained simultaneously with DEN, AE-WTN learns to transform weights from CLN to DEN through the shared classes S. The "other" detection classes (i.e., $D \setminus S$) are trained normally as conventional classification weights. Only "other" and S contribute to the detection loss ℓ_{cls}. AE-WTN uses a reconstruction loss ℓ_{rec} to reconstruct the weights for both S and N, from its encoder's outputs. *Test phase* (dashed polygon): CLN's weights of both the novel classes N and shared classes S can be adapted offline for use in DEN through AE-WTN. With that, DEN is able to detect novel classes N in addition to S and "other" classes.

originally attained by the pre-trained CLN for handling a large number of categories. We hypothesize that by letting WTN have a narrow view of the class population, its modeling capability (relating to N specifically) is severely underexploited and this compromises the performance of WTN on classes N.

To this end, we introduce Autoencoder-WTN (AE-WTN) – a novel WTN variant that attempts to preserve knowledge on all of classes C contained in pre-trained W_C, while learning a discriminative WTN function to achieve good detection performance. AE-WTN is an autoencoder with both encoder and decoder networks. AE-WTN is built on top of WTN$^+$. The encoder network shares the same architecture as WTN$^+$'s, while the decoder network (with separate network layers/parameters) is the mirrored version of the encoder. Following existing WTN, the encoder network works as a function $T()$ to predict W_D given W_C as input. During training, gradients are propagated from DEN's loss to the encoder network. The network architecture of AE-WTN and how it interacts with CLN and DEN are illustrated in Fig. 3.

AE-WTN is trained with an additional autoencoder-based training loss – *reconstruction loss* [11, 14] that forces the decoder network to predict (or reconstruct) the original inputs, from the output activations of the encoder network. Let $T()$ denote the encoder network and $G()$ denote the decoder network, the reconstruction is predicted as follows: $\hat{\mathbf{w}}_C = G(T(\mathbf{w}_C));\ \forall \mathbf{w}_C \in W_C$. Here, we adopt smooth L1 loss [9] as the reconstruction loss to minimize the difference between the predicted reconstructions and the original inputs (W_C):

$$\ell_{\text{rec}} = \begin{cases} 0.5(\hat{\mathbf{w}}_C - \mathbf{w}_C)^2, & \text{if } |\hat{\mathbf{w}}_C - \mathbf{w}_C| < 1 \\ |\hat{\mathbf{w}}_C - \mathbf{w}_C| - 0.5, & \text{otherwise} \end{cases} \quad (1)$$

Note that we apply reconstruction loss to all CLN classes C (i.e., $S \cap N$), rather than just shared classes S. On the other hand, the detection loss (box-level classification) only cares about classes S and "other" detection classes. With such formulation, we perform multi-task training based on the following mixture of training losses (excluding Region Proposal Network's [34]): $\ell_{\text{cls}} + \ell_{\text{box}} + \alpha\ell_{\text{rec}}$, where ℓ_{box} is box regression loss and α is the loss scaling hyperparameter.

Reconstruction loss penalizes intermediate network activations which do poorly to reconstruct the original weights W_{CLN}. Since AE-WTN's output W_D (weights for DEN) is a form of intermediate network activations, they are affected by the reconstruction loss and are expected to retain original class information greatly for reconstruction purpose. In contrast, existing WTN (or even WTN$^+$) is solely driven by DEN's classification loss (which may not be optimal for model generalization) and is not compelled to retain more of potentially useful class information. Reconstruction-based information preservation has been shown to help neural networks achieve better local optima [23, 45] in supervised learning. By complementing CLN's classification loss with a reconstruction loss, AE-WTN is able to learn a non-linear mapping that achieves a good balance between class/class discriminability and class information retainment. We find that this has a regularization effect on AE-WTN and it helps improve generalization performance on the fully-annotated object categories ($D \cap S$) seen during training. This observation is aligned with the findings of [23, 45] that supervised learning can be improved with autoencoders. While we apply reconstruction loss to all classes including N

(which do not have supervised annotations), [23, 45] apply the loss to only input examples with supervised annotations. Our work also resembles semi-supervised learning where reconstruction loss (autoencoder) [31, 46] is used as an auxiliary loss to exploit unlabeled data (in this work, class N are unlabeled) to improve model performance and generalization.

During the training of existing WTN, $W_{C,N}$ the weights of novel classes N, contained in W_C, is not utilized. And, classes N do not contribute to the training. Deep neural networks are generally known to eliminate task-irrelevant information of the inputs through training [36, 42]. Thus, it is likely that WTN learns to "dismiss" some class information about classes N that is unimportant to classes S but is useful for the detection of classes N. The reconstruction loss of AE-WTN addresses such a shortcoming of existing WTN by explicitly involving the novel object classes N. The rich class information in $W_{C,N}$ (which is potentially beneficial to AE-WTN's test-performance on classes N) is preserved in the intermediate network activations of AE-WTN.

5. Experiments

5.1. Implementation Details

Training and evaluation sets for seen classes D. We use the official training and validation dataset (referred to as **OI-500**) [22] from Open Images V4 Challenge which contains 500 object classes for training and evaluating DEN on classes D. The object classes in Open Images dataset are hierarchically organized and many classes are not mutually exclusive. Open Images' official evaluation metric [22], a custom version of "Average Precision (AP) @ 0.5 IoU threshold" or AP_{50} is used for evaluation on the validation set provided. We use the same Open Images training set to train baseline Faster RCNN and our WTN-based models for fair comparisons on novel classes N.

Evaluation set for novel classes. N To evaluate DEN's performance on novel classes N, we employ two evaluation datasets. The first evaluation set (**OI-57**) is a subset of Open Images V4 *complete*/non-challenge dataset containing 57 *novel* object classes and 31,061 images. The second evaluation set (**VG-200**) is set as a subset of Visual Genome [20] dataset containing 24,690 images spanning 200 high-frequency object classes which are *novel* to DEN. We adopt the same AP_{50} metric for **OI-57**. Since many object instances in Visual Genome dataset are not annotated at all, we follow the practice of [2] by using *Average Recall*/AR_{50}@100 detections per image to gauge the detection performance of DEN on this evaluation set.

Classification Network (CLN) (*source*). Prior to training WTN and DEN, a pre-trained large-scale CLN model has to be acquired. We use a publicly available ResNet-101 pre-trained on Open Images v2 [22] with 5000 object classes. It is trained with multi-label (sigmoid) classification loss given the multi-label nature of the dataset. Training resolution is 299×299. The model is trained asynchronously with 50 GPU workers and batch size 32 for 620K training steps. Incoming features to the final classification layer is 2048-dimensional.

Detection Network (DEN) (*target*). The DEN architecture in this paper is a Faster R-CNN [34] with a backbone integrating ResNet-50 [13] and Feature Pyramid Network (FPN) [25]. ResNet-50 backbone is pre-trained on ImageNet-1k [35] dataset, and its BN parameters are frozen during training of DEN. The box-level head (for box classification and regression) is a 2-layer multi-layer perceptron (MLP) with a 2048-dimensional feature and output channels. DEN is trained with mini batches of 8 images (2 images/GPU) for a total of 180K iterations. We optimize the network using SGD with momentum of 0.9 and initial learning rate of 2×10^{-2}. The network is regularized with weight decay of 1×10^{-4}. We stick closely to the original training loss functions of Faster R-CNN except for the classification loss which we replace with sigmoid binary cross-entropy, taking Open Images class hierarchy and multilabel nature into account. The training class labels are expanded [1] based on the hierarchy tree [22] given.

Weight Transfer Network (WTN). By default, WTN variants have input/feature/output channels of 2048. For Group Normalization (GN) layer in WTN$^+$and AE-WTN, we follow the same "number of groups"/#groups hyperparameter, which is set to 32 as found to be a good choice by [44]. WTN networks are trained from scratch simultaneously with DEN using AdamW [28] using default hyperparameters and weight decay of 1×10^{-4}. For AE-WTN, α is set to 20 throughout the experiments.

5.2. Comparison with related methods

To validate the effectiveness of our proposed AE-WTN model, we experimentally compare it with existing weight transfer-related methods described in the following. Note that all these methods use the same Faster R-CNN detection framework and a ResNet-50 backbone.

•**Faster R-CNN**: Vanilla Faster R-CNN [34] performs fully supervised learning on seen classes. In contrast to WTN, vanilla Faster R-CNN learns conventional classification weights which are both linear and class-specific. To detect novel classes, we employ the nearest-neighbor approach *(NN)*, taking the detections of nearest seen classes.

•**LSDA** [15]: LSDA adapts CLN's weights for detection task by learning additive class-specific biases. To make predictions for a novel class during test-time, the biases of

Method	OI-500 (Seen) AP_{50}	OI-57 (Novel) AP_{50}	VG-200 (Novel) AR_{50}
Faster R-CNN [34]	59.55	-	-
Faster R-CNN (NN)	-	28.09	49.39
LSDA [15]	59.44	25.89	51.14
LSDA (Visual Transfer) [39]	59.44	26.43	53.03
ZSD [2] with CLN weights	47.37	34.63	38.04
ZSD [2] with fastText [18]	58.39	29.51	35.09
WTN [16]	52.87	34.94	41.91
WTN^+			
▶ default model	58.82	39.28	65.60
▷ 5× weight decay	58.46	40.79	65.87
▷ activity regularizer [30]	55.86	33.47	36.26
▷ Dropout [38]	57.14	40.09	65.52
▷ reduced capacity	58.80	37.81	63.16
AE-WTN	**59.59**	**41.07**	**66.75**

Table 1. Comparison with weight transfer-related methods on evaluation datasets – OI-500 (seen classes), OI-57 (novel classes), and VG-200 (novel classes).

nearest classes are averaged and added to CLN's weight vector. The visual similarity transfer variant [39] is also included.

•**ZSD** [2]: ZSD performs zero-shot detection through pre-trained word embeddings. In a joint visual-word embedding setting, the detector learns to output visual embeddings in the words' embedding space. Here, two kinds of embeddings are considered – CLN's weights and fastText [18].

•**WTN** [16]: This corresponds to the standard (existing) WTN model that makes use of neither normalization techniques nor reconstruction loss.

•**WTN^+ variants**: Since the reconstruction loss of AE-WTN can be seen as a regularizer, we compare it with several WTN^+ variants regularized with increased weight decay ($5\times$) [21], activity regularizer (0.01) [30], Dropout (0.3) [38] on intermediate activations, and reduced network capacity (halving the number of channels in hidden layer).

The results are given in Table 5.1. We use ResNet-50 as the backbone for the vanilla Faster RCNN detector, and its AP_{50} on OI-500 is 59.55% which is mildly worse than the 60.0% achieved by the state-of-the-art SE-ResNeXt-101 detector [1]. Overall, WTN methods outperform the non-WTN methods by large margins on the novel classes (OI-57 and VG-200), due to the powerful weight transfer function learned by WTN that can generalize to many classes. Among the WTN methods, AE-WTN that incorporates all the proposed improvements achieves the best results.

WTN and WTN^+ suffer from the weakened performance on OI-500 (seen classes D) compared with the vanilla Faster R-CNN detector that it is built upon. In other words, switching to WTN from *conventional classification weights* decreases performance on the seen classes. This phenomenon has been observed by prior works [15, 19] attempting to scale object detection with weak or partial supervision. By integrating autoencoder into WTN (AE-WTN), the seen-class detection performance can be recovered. It is extremely challenging to train conventional WTN from scratch. The reconstruction loss (which is more easily optimized than detection loss) encourages AE-WTN to output weights highly representative of original CLN weights, thus providing a good initialization to attain better local optima. Similar to prior works that find reduced supervised training loss with autoencoder [23, 45], we find that the box-level classification training loss ℓ_{cls} on seen classes attained by AE-WTN (0.5572) is lower than that of WTN^+ (0.5754).

Moreover, the reconstruction loss explicitly involves novel classes N during training and forces AE-WTN to preserve rich class information of the novel classes in the latent and output spaces. It also encourages visual features learned by DEN to be more "generic" (less specific to classes D in the detection dataset) in order to accommodate to the many classes represented by AE-WTN. Therefore, the detector equipped with AE-WTN shows improved (absolute) performances of 1.8% and 1.1% over WTN^+ on OI-57 and VG-200 respectively. Compared to other existing regularization techniques applied to WTN^+, AE-WTN performs better across all datasets. This provides confirmation that the advantages of the reconstruction loss cannot be simply replicated by other regularizers that do not leverage the rich class information contained in CLN's weights.

Qualitative results. We provide in Fig. 5 some qualitative results obtained by our proposed AE-WTN detector on test images of Open Images [22] and Visual Genome [20] datasets. Only the classes with the highest scores are shown, and novel classes compete with seen classes for the same bounding box. Remarkably, the detector can detect a variety of novel classes at greater confidence than seen classes, despite not seeing them during training.

5.3. Analysis

Local neighborhood preservation. To better understand the implications of the reconstruction loss on local neighborhood preservation of AE-WTN, we compute the overlapping count between nearest neighbors obtained by CLN's weights and the output weights of the WTN model of interest (AE-WTN, WTN^+, or WTN), varying the number of neighbors (a standard hyperparameter of *nearest neighbor* approach) for all methods. This study is performed on 20 randomly-sampled classes and the counts are averaged across those classes. Nearest neighbors are among the 5,000 classes of CLN. The findings are presented in Fig. 4. E.g., at 100 neighbours, AE-WTN's output weights and CLN's weights have an average of 48.25 overlapping neighbours, while WTN^+ and WTN have 38.0 and 31.95 overlapping neighbours respectively. As shown, AE-WTN consistently reaches greater numbers

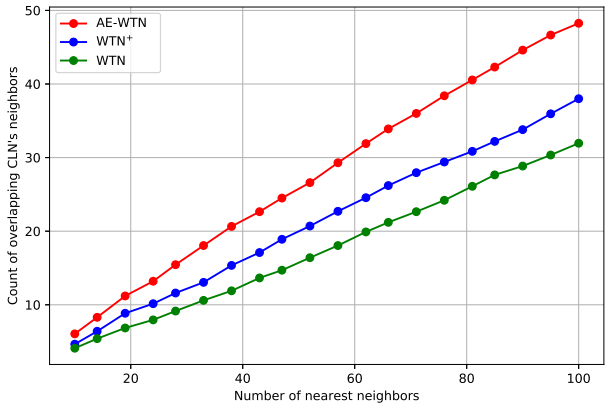

Figure 4. The overlapping count (vertical axis) between CLN's nearest neighbors and the nearest neighbors obtained by the WTN model of interest (AE-WTN, WTN$^+$, or WTN), given varying numbers of nearest neighbors (horizontal axis).

WTN→WTN$^+$		OI-500 (Seen)	OI-57 (Novel)	VG-200 (Novel)
Input Norm.	**Group Norm.**	AP_{50}	AP_{50}	AR_{50}
✗	✗	52.87	34.94	41.91
✓	✗	57.60	37.27	54.19
✗	✓	54.60	35.84	58.55
✓	✓	**58.82**	**39.28**	**65.60**

Table 2. Ablation study on WTN$^+$architecture.

of overlapping neighbors (with CLN's neighbors) than WTN$^+$and WTN do, indicating that AE-WTN can better preserve the local neighborhood relationships of classes than WTN$^+$. Noticeably, the gap widens as the number of nearest neighbors increases.

Normalizations in WTN$^+$. We perform ablation study in Table 2 to understand how the performance changes with different normalization techniques. It is crucial to combine the two normalizations of WTN$^+$to obtain the best results for both seen and novel classes. Furthermore, we observe worse training losses with non-normalized WTN compared with WTN$^+$, implying that model under-fitting is the inherent cause of WTN's under-performance.

Choice of feature normalization. GN [44] is chosen over the typical BatchNorm (BN) [17] because for WTN$^+$, BN is less robust towards novel-class inputs which do not have detection annotations/loss [8]. We find that the post-ReLU activation (L2) norms of WTN$^+$with BN have an unusually large *variance* for novel classes. It is 70× (or $\frac{7.117}{0.104}$) that of shared classes, despite allowing BN to normalize over all classes in training. Such unstable activations are not encountered by the detection network during training. This

	mean		variance	
	shared cls.	**novel cls.**	**shared cls.**	**novel cls.**
GN [44]	1.838	1.784	0.091	0.093
BN [17]	1.379	**2.627**	0.104	**7.117**

Table 3. *Means* and *variances* of post-ReLU activation norms.

	Faster R-CNN	**WTN**	**WTN$^+$**	**AE-WTN**
Time (ms)	365	371	379	401
Mem. (GB)	4.11	4.15	4.19	4.26

Table 4. Training time and memory usage.

causes WTN$^+$'s predicted weights for novel classes to interact poorly with image-region features at test time, resulting in unreliable class-score predictions. Table 3 shows the L2 norm *means* & *variances* of using GN and BN.

Computational efficiency. Computational efficiency is a major concern in the training and/or deployment of object detectors, especially for large-scale detectors. In Table 4, we show the per-iteration training time (in milliseconds/*ms*) and single-GPU memory usage of training with different models. Overall, the WTN models add very little computational costs on top of Faster R-CNN's. During testing, all the weights can be transformed offline with WTN/WTN$^+$/AE-WTN to reach vanilla Faster R-CNN's efficiency.

6. Conclusion

Training large-scale object detectors is extremely resource-demanding (e.g., data, computations). In this work, we introduce an efficient and effective WTN approach to scale up object detection, and propose novel methods to strongly push the limits of WTN through normalization techniques and autoencoder-based reconstruction loss. The reconstruction loss adopted by AE-WTN effectively improves its capability to retain and exploit the semantically-rich class information (of all classes) learned by the pre-trained CLN. This leads to improved training of DEN and better detection performances on both seen and novel classes.

Acknowledgement

Jason Kuen was supported by NTU Research Scholarship under School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore.

References

[1] Takuya Akiba, Tommi Kerola, Yusuke Niitani, Toru Ogawa, Shotaro Sano, and Shuji Suzuki. Pfdet: 2nd place solution to open images challenge 2018 object detection track. *arXiv preprint arXiv:1809.00778*, 2018. 5, 6

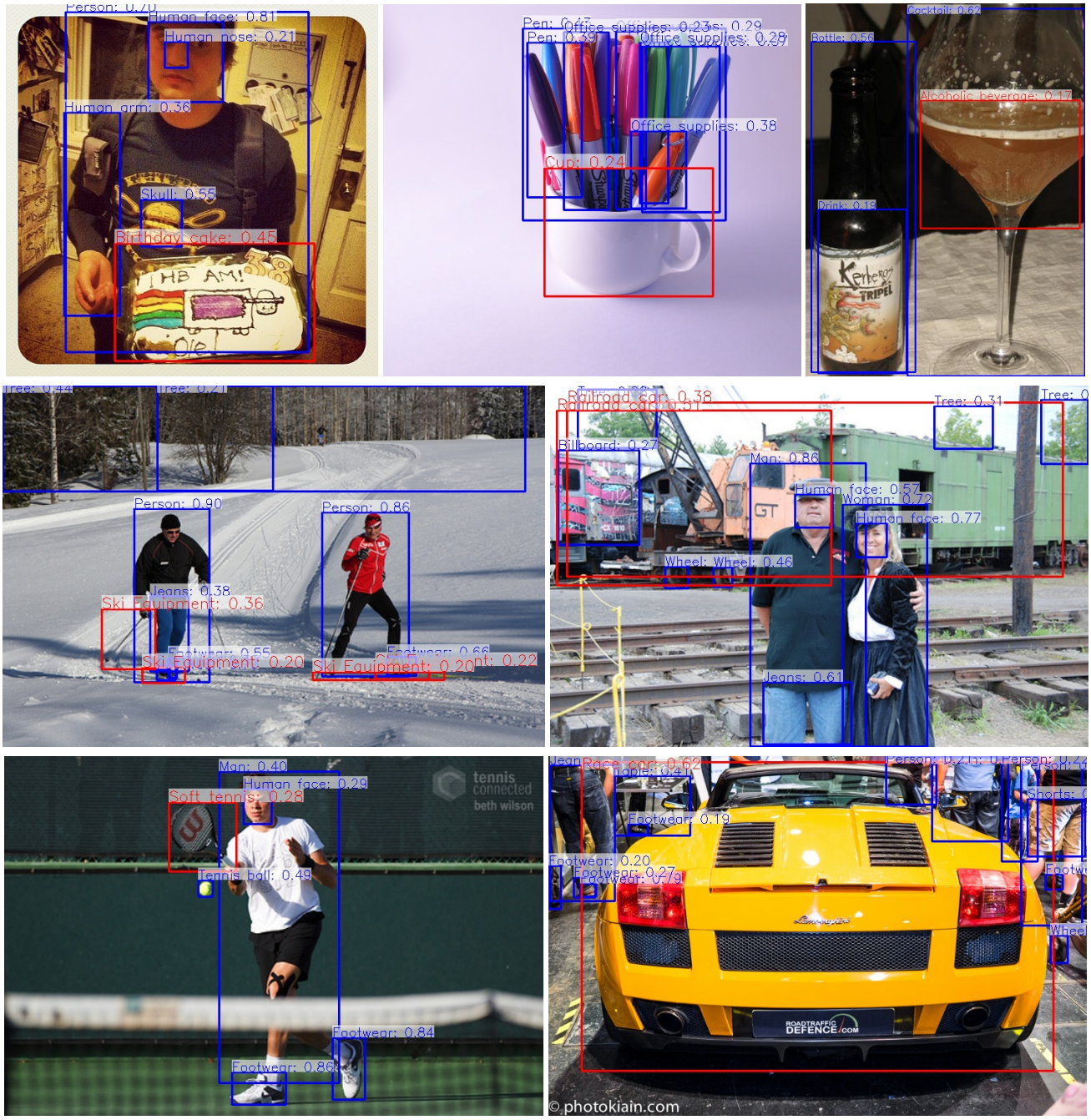

Figure 5. Qualitative results of the proposed AE-WTN. The blue detection boxes **seen** object classes D, and the red boxes labels are **novel** classes N. Our method can handle a variety of novel classes and concepts, while performing well on seen classes.

[2] Ankan Bansal, Karan Sikka, Gaurav Sharma, Rama Chellappa, and Ajay Divakaran. Zero-shot object detection. In *ECCV*, 2018. 5, 6

[3] Hakan Bilen, Marco Pedersoli, and Tinne Tuytelaars. Weakly supervised object detection with convex clustering. In *CVPR*, pages 1081–1089, 2015. 1

[4] Hakan Bilen and Andrea Vedaldi. Weakly supervised deep detection networks. In *CVPR*, pages 2846–2854, 2016. 1

[5] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. In *NIPS*, pages 379–387, 2016. 2

[6] Ramon Lopez De Mantaras and Eva Armengol. Machine learning from examples: Inductive and lazy methods. *Data & Knowledge Engineering*, 25(1-2):99–123, 1998. 3

[7] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. 1

[8] Angus Galloway, Anna Golubeva, Thomas Tanay, Medhat

Moussa, and Graham W Taylor. Batch Normalization is a Cause of Adversarial Vulnerability. In *ICML Workshop*, 2019. 7

[9] Ross Girshick. Fast r-cnn. In *ICCV*, pages 1440–1448, 2015. 2, 4

[10] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, pages 580–587, 2014. 2

[11] Jiuxiang Gu, Handong Zhao, Zhe Lin, Sheng Li, Jianfei Cai, and Mingyang Ling. Scene graph generation with external knowledge and image reconstruction. In *CVPR*, 2019. 4

[12] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, pages 2980–2988. IEEE, 2017. 1

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 5

[14] Geoffrey E Hinton and Richard S Zemel. Autoencoders, minimum description length and helmholtz free energy. In *NIPS*, pages 3–10, 1994. 4

[15] Judy Hoffman, Sergio Guadarrama, Eric S Tzeng, Ronghang Hu, Jeff Donahue, Ross Girshick, Trevor Darrell, and Kate Saenko. Lsda: Large scale detection through adaptation. In *NIPS*, pages 3536–3544, 2014. 2, 5, 6

[16] Ronghang Hu, Piotr Dollár, Kaiming He, Trevor Darrell, and Ross Girshick. Learning to segment every thing. In *CVPR*, pages 4233–4241, 2018. 1, 2, 6

[17] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pages 448–456, 2015. 3, 7

[18] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. In *ACL*, pages 427–431. Association for Computational Linguistics, April 2017. 6

[19] Bingyi Kang, Zhuang Liu, Xin Wang, Fisher Yu, Jiashi Feng, and Trevor Darrell. Few-shot object detection via feature reweighting. *arXiv preprint arXiv:1812.01866*, 2018. 6

[20] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *IJCV*, 123(1):32–73, 2017. 2, 5, 6

[21] Anders Krogh and John A Hertz. A simple weight decay can improve generalization. In *NIPS*, pages 950–957, 1992. 6

[22] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Tom Duerig, and Vittorio Ferrari. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *arXiv:1811.00982*, 2018. 1, 2, 5, 6

[23] Lei Le, Andrew Patterson, and Martha White. Supervised autoencoders: Improving generalization performance with unsupervised regularizers. In *NeurIPS*, pages 107–117, 2018. 4, 5, 6

[24] Dong Li, Jia-Bin Huang, Yali Li, Shengjin Wang, and Ming-Hsuan Yang. Weakly supervised object localization with progressive domain adaptation. In *CVPR*, pages 3512–3520, 2016. 1

[25] Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 2117–2125, 2017. 2, 5

[26] 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*, pages 740–755. Springer, 2014. 1

[27] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *ECCV*, pages 21–37. Springer, 2016. 2

[28] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 5

[29] Dhruv Mahajan, Ross B. Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. In *ECCV*, 2018. 1

[30] Stephen Merity, Bryan McCann, and Richard Socher. Revisiting activation regularization for language rnns. *arXiv preprint arXiv:1708.01009*, 2017. 6

[31] Antti Rasmus, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. Semi-supervised learning with ladder networks. In *NIPS*, pages 3546–3554, 2015. 5

[32] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, pages 779–788, 2016. 2

[33] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. In *CVPR*, pages 6517–6525. IEEE, 2017. 1, 2

[34] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: towards real-time object detection with region proposal networks. *TPAMI*, (6):1137–1149, 2017. 1, 2, 4, 5, 6

[35] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015. 5

[36] Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *arXiv preprint arXiv:1703.00810*, 2017. 5

[37] Bharat Singh, Hengduo Li, Abhishek Sharma, and Larry S Davis. R-fcn-3000 at 30fps: Decoupling detection and classification. In *CVPR*, pages 1081–1090, 2018. 2

[38] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 15(1):1929–1958, 2014. 6

[39] Yuxing Tang, Josiah Wang, Boyang Gao, Emmanuel Dellandréa, Robert Gaizauskas, and Liming Chen. Large scale semi-supervised object detection using visual and semantic knowledge transfer. In *CVPR*, pages 2119–2128, 2016. 1, 2, 6

[40] Yuxing Tang, Josiah Wang, Xiaofang Wang, Boyang Gao, Emmanuel Dellandréa, Robert Gaizauskas, and Liming Chen. Visual and semantic knowledge transfer for large scale semi-supervised object detection. *TPAMI*, 40(12):3045–3058, 2018. 1, 2

[41] Qingyi Tao, Hao Yang, and Jianfei Cai. Zero-annotation ob-

ject detection with web knowledge transfer. In *ECCV*, 2018. 2

[42] Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *IEEE Information Theory Workshop (ITW)*, pages 1–5. IEEE, 2015. 5

[43] Jasper Uijlings, Stefan Popov, and Vittorio Ferrari. Revisiting knowledge transfer for training object class detectors. In *CVPR*, 2018. 2

[44] Yuxin Wu and Kaiming He. Group normalization. In *ECCV*, pages 3–19, 2018. 3, 5, 7

[45] Yuting Zhang, Kibok Lee, and Honglak Lee. Augmenting supervised neural networks with unsupervised objectives for large-scale image classification. In *ICML*, pages 612–621, 2016. 4, 5, 6

[46] Junbo Zhao, Michael Mathieu, Ross Goroshin, and Yann Lecun. Stacked what-where auto-encoders. *ICLR*, 2016. 5