Generic Perceptual Loss for Modeling Structured Output Dependencies

Yifan Liu[1], Hao Chen[1], Yu Chen[2], Wei Yin[1], Chunhua Shen[1,3*]
[1] The University of Adelaide [2] Automind [3] Monash University, Australia

Abstract

The perceptual loss has been widely used as an effective loss term in image synthesis tasks including image super-resolution [16], and style transfer [14]. It was believed that the success lies in the high-level perceptual feature representations extracted from CNNs pretrained with a large set of images. Here we reveal that, what matters is the network structure instead of the trained weights. Without any learning, the structure of a deep network is sufficient to capture the dependencies between multiple levels of variable statistics using multiple layers of CNNs. This insight removes the requirements of pre-training and a particular network structure (commonly, VGG) that are previously assumed for the perceptual loss, thus enabling a significantly wider range of applications. To this end, we demonstrate that a randomly-weighted deep CNN can be used to model the structured dependencies of outputs. On a few dense per-pixel prediction tasks such as semantic segmentation, depth estimation and instance segmentation, we show improved results of using the extended randomized perceptual loss, compared to the baselines using pixel-wise loss alone. We hope that this simple, extended perceptual loss may serve as a generic structured-output loss that is applicable to most structured output learning tasks.

1. Introduction

Dense pixel-wise prediction tasks represent the most important category of computer vision problems, ranging from low-level image processing such as denoising, super-resolution, through mid-level tasks such as stereo matching, to high-level understanding such as semantic/instance segmentation. These tasks are naturally structured output learning problems since the prediction variables often depend on each other. The pixel-wise loss serves as the unary term for these tasks. Besides, the perceptual loss [14] was introduced to capture perceptual information by measuring discrepancy in high-level convolutional features extracted from CNNs. It has been successfully used in various low-level image processing tasks, such as style transfer, and super-resolution [14].

Previous works assume that the perceptual loss benefits from the high-level perceptual features extracted from CNNs pretrained with a large set of images (*e.g.*, VGG [23] pretrained on the ImageNet dataset. Relying on this assumption, the perceptual loss is limited to *a specific network structure* (commonly, VGG) with *pre-trained* weights, which is not able to take arbitrary signals as the input. In this work, we reveal that, contrary to this belief, the success of the perceptual loss is not necessarily dependent on the ability of a *pretrained* CNN in extracting high-level perceptual features. Instead, *without any learning*, the structure of a multi-layered CNN is sufficient to capture a large amount of interaction statistics for various output forms. We argue that what matters is the deep network architecture rather than the pretrained weights.

To verify the statement, we conduct a pilot experiment on image super-resolution. Apart from using the pretrained VGG net for perceptual loss, we use a randomly-weighted network. The results with the randomly-weighted network are *on par* with that of the pretrained VGG, which are both visually improved than using the per-pixel loss alone (see Figure 1). This indicates that the pretrained weights—previously assumed for the perceptual loss—is not essential to the success of the perceptual loss. We may conclude from this experiment that it is the deep network structure, rather than learnt weights, plays the core role.

Given a target y or a prediction $\hat{y}$ as an input, a randomly-weighted network $f(\cdot)$ can work as a function to explore hierarchical dependencies between variable statistics through the convolution operations in multiple layers. Thus, a generic perceptual loss for structured output learning can be computed by comparing the discrepancy between $f^j(y)$ and $f^j(\hat{y})$. Here j indexes a particular layer of the network $f(\cdot)$. Thus, this enables the perceptual loss[1] to be applied to a wider range of structured output learning tasks.

Structured information is important in dense per-pixel prediction problems, such as semantic segmentation [17],

*C. Shen is the corresponding author.

[1]Here we still use the notion of 'perceptual' as it was firstly introduced in [14] even though broadly this loss is more about capturing inter-dependencies in variables, instead of extracting perceptual features.

(a) Ground Truth **(b)** Bicubic **(c)** Pixel-wise Loss alone [16] **(d)** w. Pretrained VGG [16] **(e)** w. Random VGG

Figure 1 – Super-resolution results of the pilot experiments (super-resoled from 4× down-scaled images). (a) Ground truth high-resolution images. (b) Bi-cubic up-sampling. (c) SRResNet [16] trained with the per-pixel loss. (d) SRResNe trained with the per-pixel loss and perceptual loss with a pretrained VGGNet. (e) SRResNet trained with the per-pixel loss and perceptual loss with a randomly-weighted VGGNet. We can see that the perceptual loss improves image quality. Besides, formulating the perceptual loss with a pre-trained network and a randomly-weighted network produces *on par* results.

depth estimation and instance segmentation [18]. For example, the pairwise term in Markov Random Filed is complementary to the unary term, which defines pairwise compatibility and in general improves prediction accuracy especially when the unary term alone is not sufficient. The proposed generic perceptual loss can be easily applied to these dense prediction tasks, with *no computation overhead* during inference. Also, as now pre-training with labelled data is not required, it is straightforward to explore the effectiveness of using various network structures—not necessary the VGG—to model the dependency between output variables.

Experimental results on various structured output learning tasks with different network structures show that the generic perceptual loss benefits the training, and consistently achieves improved performance compared to the baselines using pixel-wise loss term alone. We also provide detailed comparisons and analysis on the impact of initialization schemes and architectures of the perceptual loss network.

In summary, our main contributions are as follows.

- We reveal that the success of the perceptual loss is not dependent on the pretrained CNN weights. Without any learning, the structure of a deep network is sufficient to capture the dependencies between multiple levels of variable statistics using multiple layers of CNNs.
- We apply the generic perceptual loss to a few structured output learning tasks, including semantic segmentation, depth estimation and instance segmentation. We consistently improve the performance of baseline models.
- We investigate how the initialization and the network structures may affect the performance of the proposed perceptual loss. A reliable initialization approach is designed based on the analysis.
- This proposed simple perceptual loss may serve as a generic structured-output loss that is applicable to most structured output learning tasks in computer vision.

2. Related Work

Perceptual loss. Early works [16, 14] generate high-quality images using perceptual loss functions, which consider the discrepancy between deep features, not only the pixels. Gatys *et al*. [8] found that a pretrained VGG architecture can be used to as a loss function for style transfer. They attribute the success to the ability of the trained filters in learning certain features which are coincident to human perception. Johnson *et al*. [14] further formulate the perceptual loss as an extra loss term for the deep neural network. There, the context/perceptual loss is the Euclidean distance between feature representations. Inclusion of the percep-

tual loss achieves visually improved results on style transfer and super-resolution. While doing other image synthesis tasks, such as super-resolution, colorizing, and other image generation tasks [13], the perpetual loss often refers to this context loss as shown in Eq. (1),

$$\ell_{feat}^{\phi,j}(\hat{y}, y) = \frac{1}{C_j H_j W_j} \|\phi_j(\hat{y}) - \phi_j(y)\|_2^2, \qquad (1)$$

where y and $\hat{y}$ are the targeted images and synthesis images. ϕ_j represents the perceptual function which outputs the activation of the jth layer in the perceptual loss network. C_j, H_j, W_j are the dimensions of the tensor feature map.

Recently, a perceptual loss was also introduced to depth estimation task [31]. The authors argue that the embedding spaces should be designed for particular relevant tasks, *i.e.*, depth-based scene classification and depth reconstruction. Thus, they have pretrained the perceptual loss network on RGBD datasets and failed to realize that pretraining is not compulsory, as we show here.

Representations with random weights. A few methods have discussed the untrained, randomly-weighted CNNs. Researchers found that networks with random weights can extract useful features as the classification accuracy with these features is higher than random guesses. He *et al*. [9] use generative models with the constraints from untrained, randomly-weighted network for deep visualization tasks. They found that during optimization for a style transfer task, the perceptual loss with a proper weight scale can work well with untrained, randomly-weighted networks, and generate competitive results as prior work [8] with pretrained weights. Mongia *et al*. [19] prove why one-layer CNNs with random weights can successfully generate textures. The randomly-weighted networks are also employed in unsupervised learning [29] and reinforcement learning [7]. Another relevant work is deep image prior [28] where a randomly-initialized neural network is used as a "handcrafted prior" with excellent results in image reconstruction tasks such as denoising, super-resolution, and inpainting. These works provide a way to study network architectures without any learning, and also exploit the randomness as a useful feature.

Here, we explore the ability of the randomly-weighted network in investigating hierarchical dependencies between variable statistics. The perceptual loss with a randomly-weighted network works as a useful loss term on various structured output learning tasks.

Dense prediction. Dense prediction is a family of fundamental problems in computer vision, which learns a per-pixel mapping from input images to output structures, including semantic segmentation [37], depth estimation [35, 33, 36], object detection [27], *etc.* As extensively studied in the literature, taking the inter-dependency between output variables into account during training and/or inference often improves the accuracy. Thus structured information is important for these tasks.

In this work, we demonstrate that a randomly-weighted network can implicitly capture the structural information with its natural architecture and internal convolution operations. The performance of these dense prediction tasks can be enhanced by simply enforcing a perceptual loss from a randomly-weighed network. This is achieved *without any learning* on the perceptual loss network, and *no further computation cost* for inference is required.

3. Our Method

The observations from the pilot experiments in Figure 1 suggest that the network structure, instead of the pretrained weights, contributed to the success of the perceptual loss. More details can be found in the supplementary materials. In this section, we first extend the perceptual loss with randomly-weighted networks to some structured output learning tasks. Then, we analyze the devils in the weight initialization and design an appropriate way for an effective initialization.

3.1. Perception Loss for Structured Output Prediction

If the randomly-weighted network has the ability in capturing structured information, it should also be able to help dense prediction problems. In addition, as the pretrained weights learnt with a large number of samples are not required, it is easier to apply this regularization on any task and to compare the performance with different perceptual loss networks. We start with the commonly used VGGNet-like structure as the perceptual loss network . We denote the number of convolutional layers between max pooling downsample operations with $N_1, N_2, \ldots, N_k$, where k is the number of blocks.

Semantic segmentation. Semantic segmentation is a typical dense prediction problem, where a semantic label is assigned to each pixel in an input image. A segmentation network as $\mathcal{S}$ takes an input image $\mathbf{I} \in \mathbb{R}^{W \times H \times 3}$ and predicts a segmentation map $\hat{y} = \mathcal{S}(\mathbf{I}) \in \mathbb{R}^{W \times H \times C}$. The output channel of the segmentation network C equals to the number of the pre-define object classes. Conventional methods usually employ a per-pixel cross-entropy loss. The correlations among pixels are neglected in the cross-entropy loss. Therefore, the perceptual loss can work as a complementary to the per-pixel loss for capturing the structured information.

To extend the perceptual loss to semantic segmentation, we use the estimated segmentation map or the ground-truth one as the input to the perceptual loss network, and get the embedded structured feature after several CNN layers. The mean square error is used to minimize the distance between the structured features of prediction and the learning target.

The softmax output $\hat{y}$ has a domain gap with the one-hot ground truth y, which make the perceptual loss hard to converge. To solve this problem, we follow recently knowledge distillation methods [18] to generate soft labels y_t by a large teacher net as the learning targets. The total loss is then defined as:

$$\ell_{\text{seg}} = \ell_{ce}(\hat{y}, y) + \lambda \cdot \ell_r^{\phi_r}(\hat{y}, y_t), \tag{2}$$

where ϕ_r represents the perceptual loss network initialized with random weights and we set λ as 0.1 in all experiments.

Depth estimation. Monocular depth prediction [35, 6] is a regression problem, which predicts the per-pixel real-world distance from the camera imaging plane to the object captured by each pixel in a still image $\mathbf{I}$. We use VNL proposed by Yin *et al*. [35] as a baseline model. The pixel-level depth prediction loss and the virtual normal loss are used to supervise the network. The network outputs a predicted depth map $\hat{d} \in \mathbb{R}^{W \times H \times 1}$. Therefore, the input channel of the perceptual loss net equals to 1. As the ground-truth depth map follows the same statistical distribution as the estimations, the target and the estimation can be directly used as inputs to the perceptual loss network. The network with virtual normal loss, as a strong baseline, minimizes the difference of a manually defined geometry information between the prediction and the ground truth, *i.e.*, the direction of the normal recovered with three samples. The perceptual loss can still capture extra structured information when combined with ℓ_{vn}.

Instance segmentation. Instance segmentation is one of the most challenging computer vision tasks, as it requires the precise per-pixel object detection and semantic segmentation simultaneously. Recently, one stage methods achieve promising performance [26, 2, 34], making the pipeline more elegant and easier to implement. The mask and classification logits are predicted for each pixel in the feature space.

We follow CondInst [26], a state-of-the-art one-stage method, as a strong baseline. In the training procedure, CondInst consists of a bounding box detection head and a mask head. The detection head also includes a controller, which is dynamically applied to the mask head for different instances. In this way, it will produce single channel segmentation masks for each instance in the training batch with the shape of $1/4W \times 1/4H$. If there are n predicted instances in the training batch, the input of the perceptual loss network is then $n \times 1 \times 1/4W \times 1/4H$. Similar to semantic segmentation, the soft targets are generated by a teacher network.

3.2. Devils in the Initialization

The initialization of the randomly-weighted network affects the performance of the generic perceptual loss. As we employ the structured predictions as the input to the perceptual loss network and produce an embedding, an inappropriate initialization may lead to an unstable results. It is important to guarantee that each layer is bounded by a Lipschitz constant close to 1, so that the gradient generated by the random network will not explode or vanish. Here we investigate the gradient scales in the random network and derive a robust initialization method by following [10]. For a dense prediction task, we have the prediction results (*e.g.*, segmentation map) $\mathbf{Y}'$, and also its ground-truth or soft targets $\mathbf{Y}$. Our random-weight perceptual loss network transforms $\mathbf{Y}'$ and $\mathbf{Y}$ into two embeddings $\mathbf{E}'$ and $\mathbf{E}$. The generic perceptual loss is computed as the discrepancy between $\mathbf{E}'$ and $\mathbf{E}$,

$$\ell_r = \|\mathbf{E}' - \mathbf{E}\|_2^2. \tag{3}$$

Suppose that $\mathbf{e}_l$ is the response activation values related to the corresponding $k \times k$ pixels in the convolution operation with kernel size k. We have:

$$\mathbf{e}_l = \mathbf{W}_l \mathbf{y}_l + \mathbf{b}_l. \tag{4}$$

Here, l is the index of a layer. $\mathbf{y}_l$ is the input vector with $n_l = k^2 c_l$ elements, where c_l is the input channel of this layer. $\mathbf{W}_l$ is a d_l-by-n_l matrix, where d is the number of random initialized filters in this layer. With a deep convolutional network, we have $\mathbf{y}_l = f(\mathbf{e}_{l-1})$, where $f(\cdot)$ is the ReLU activation function. We also have $c_l = d_{l-l}$. If we initialize w_l with a symmetric distribution around zero and $b_l = 0$, then e_l has zero mean and has a symmetric distribution around zero. Following [10], we compute the variance of the output embedding after L layers:

$$Var[e_L] = Var[e_1] \left(\prod_{l=2}^{L} \frac{1}{2} n_l Var[w_l] \right). \tag{5}$$

The variance of the discrepancy is upper bounded by the same factor:

$$\begin{aligned} &Var[(e'_L - e_L)] \\ &= Var[e'_L] + Var[e_L] - 2Cov[e'_L, e_L] \\ &\leq Var[e_l] + Var[e'_l])(\prod_{l=2}^{L} \frac{1}{2} n_l Var[w_l]. \end{aligned} \tag{6}$$

When L becomes extremely large, the product $\prod_{l=2}^{L} \frac{1}{2} n_l Var[w_l]$ vanishes or explodes if $\frac{1}{2} n_l Var[w_l] \neq 1$. Therefore we initialize each layer using a zero-mean Gaussian distribution with a standard deviation (std) of $\sqrt{2/n_l}$ as in [10]. Before the optimization of the task network, the correlation between Y and Y' is very small. Therefore, the scale of the θ is small. In the training process, the weights of the random network ($\mathbf{W}$) are fixed, but Y' becomes closer to Y as the task network is optimized. Then the covariance becomes close to the variance of the two embeddings and this bound tends to zero.

4. Experiments

In this section, we first investigate some interesting questions about the perceptual loss network, and then employ an efficient and effective structure as the perceptual loss network to show its ability in boosting performance in a few dense prediction tasks, including semantic segmentation, depth estimation and instance segmentation.

4.1. Discussions

We share some observations in exploring the capacity of the random weight perceptual loss networks. We ask a few questions including: *Will the trained filters help the perceptual loss in dense prediciton problems? How does the depth/receptive field/multi-scale losses affect the performance? How does the initialization affect the performance?* Discussions are base on semantic segmentation task with Cityscapes [4] as the training set. PSPNet [38] with Resent18 [11] as the backbone is used as a baseline model, which is trained with the per-pixel cross-entropy loss. The soft targets are generated by the PSPNet with Resent101 [11] as the backbone. The training settings follow the details in Section 4.2.1. The performance is evaluated on the validation set of Cityscapes with the mean of Intersection over Union (mIoU) as the metric.

P Net	R: mIoU (%)	T: mIoU (%)
Non-VGG families		
GoogleNet [25]	68.91	68.90
AlexNet [15]	69.80	69.87
MobileNetV2 [22]	69.98	70.01
ResNet18 [11]	70.16	70.14
VGG families		
VGG16 [23]	70.68	70.71
VGG19 [23]	**71.25**	71.19

Table 1 – Results of the perceptual loss for semantic segmentation with a few different networks. 'R' indicates that we randomly weight the loss network. 'T' means that we assign the network with the pretrained kernels from ImageNet classification. The baseline model achieves 69.60% of mIoU.

4.1.1 Training Weights vs. Architecture

The pretrained filters were considered the key to the success of the perceptual loss. Apart from the visualization results on image super-resolution in the previous pilot experiment, we show quantitative analysis on structured output learning tasks in this section.

Taking the semantic segmentation task as an example, we use the same semantic segmentation network and training settings in our experiments, and only change the perceptual loss networks. We assign the weights of the perceptual loss network with pretrained kernels to see if the trained filters can help improve the performance. Also, we choose different network structures as the perceptual loss network, including VGG families [23], ResNet18 [11], GoogleNet [25], AlexNet [15] and MobileNetV2 [22]. The results are shown in Table 1. 'R' means that the weighs of the perceptual loss network are randomly initialized following Section 3.2. 'T' means that we employ the per-trained weights on the ImageNet to initialize the perceptual loss network, and the dense prediction output is transferred from C channels to 3 using a 1×1 convolutional layer to fit the pretrained network structure.

Percep. network	Structure	R: mIoU (%)	T: mIoU (%)
N/A	1, 1, 1, 1, 1	70.88 ± 0.03	N/A
VGG11	1, 1, 2, 2, 2	70.18 ± 0.11	70.21 ± 0.10
VGG13	2, 2, 2, 2, 2	70.64 ± 0.14	70.62 ± 0.12
VGG16	2, 2, 3, 3, 3	70.68 ± 0.03	70.71 ± 0.02
VGG19	2, 2, 4, 4, 4	71.25 ± 0.04	71.19 ± 0.07
N/A	3, 3, 4, 4, 4	70.89 ± 0.23	N/A

Table 2 – The perceptual loss with variants of VGG as the perceptual loss network (kernel size: 3). We vary the number of convolutional layers in each block. 'R' means random initialization. 'T' means initialization with pre-trained weights

From the table, we can see that training with random weights and the pre-trained weights show almost no difference on improvements (difference around 0.02% to 0.06%), but different network structures lead to a larger performance gap (varies from 68.90% to 71.25%).

This indicates that , in this structured output learning task, the trained filters for ImageNet is not the key to the success of perceptual loss. Meanwhile, the network architecture of the perceptual loss network affects the ability of capturing the structured information.

Interestingly, we observe that the VGG families perform better than other structures. VGG families have been shown unique in a few previous works. Researchers find that in style transfer, the VGG structure can work better than ResNet [12, 20], and explain the reason as the VGGNet is more robust than the ResNet. Su *et al*. [24] observe that VGG families exhibit high adversarial transfer-ability than other structures. However, the theoretical explanation of why VGG structures show a better performance is still not fully investigated in literature.

An independent recent work [31] for depth estimation also shows interesting results by employing a convolutional network to map the structured output into an embedding space. They argue that if the perceptual loss network are trained with some highly related tasks, the performance is improved. So they require additional annotations and try to design an efficient network structure as the perceptual loss network. Different from their work, we focus on a more general discussion on the perceptual loss, and verify that the network architecture plays a more important role than

pretrained weights. Although training the perceptual loss network on highly relative tasks with additional information may further improve the performance, it is not generic to be applied to various tasks and requires extra annotation effort.

4.1.2 Design of the Perceptual Loss Network

In the previous section, we have shown that the network architecture plays a more important role in the perceptual loss. As our method does not require pre-training, it is convenient to investigate the performance with various network structures. Here we explore different designs for the randomized perceptual loss network. All the experiments are conducted for three times with random initialization, and we report the mean and derivation for each setting.

Impact of depth. Conventional VGG families have variants with different numbers of layers, such as the most popular VGG16 and VGG19. These VGG families contains five convolutional blocks, and a max pooling layer at the end of each block. For each block, the feature dimension is $64, 128, 256, 512, 512$. We change the number of convolutions in each block, as shown in the 'Structure' in Table 2. For the typical structures in original VGG families, we also report the results with pretrained weight on ImageNet as in Section 4.1.1. All the convolutions' kernerl size is 3 as in VGGNet. From Table 2, we can find a consistent conclusion that pretrained kernels do not help to improve the performance, while the network structures exhibit a larger impact. With the kernel size of 3, VGG19 is the most effective perceptual loss network. Besides, the structure of '$1, 1, 1, 1, 1$' (only 1 conv. layer for each block) also show good performance. As the training memory and the training time may increase as the perceptual loss network becomes deeper, the structure of '$1, 1, 1, 1, 1$' among these settings is the best choice considering both effectiveness and efficiency.

Impact of the receptive field. The randomly-weighted network can capture the structured correlation among local features, as the convolution operation has the ability to exploit the information at multiple scales of receptive fields. We conduct experiments to see if a larger receptive field can help the randomly weighted network to better capture the structured information. We employ the structure of '$1, 1, 1, 1, 1$' as the basic perceptual loss network and adjust the kernel size in each convolutional layer to change the receptive field. The results are shown in Table 3. We can see that a larger kernel size might lead to slightly better performance on average (from 70.71% to 71.16%). Note that the derivation of the results increases (from 0.02% to 0.12%).

We plot the pixel accuracy and mIoU on the validation dataset w.r.t. the training iterations in Figure 2. Clearly the perception loss helps the training and almost during the entire training course, we observe improved performance when the perception loss is used.

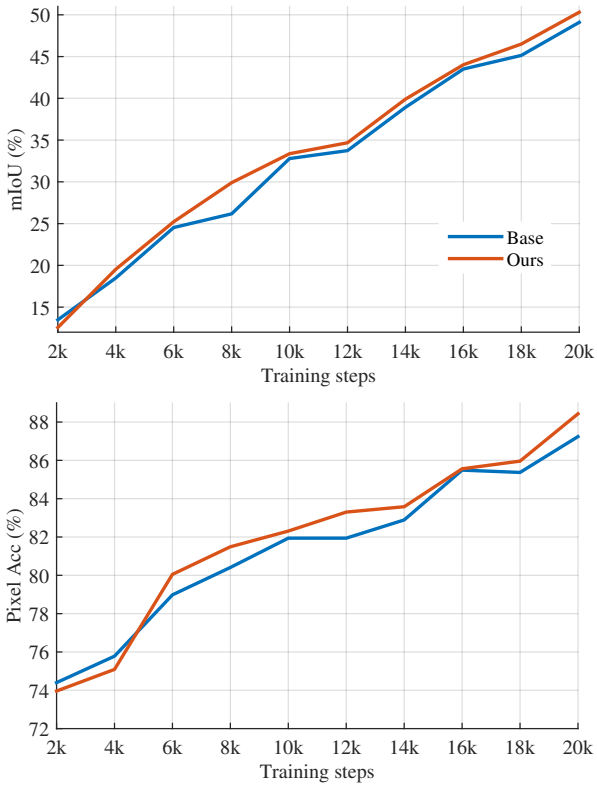

Figure 2 – Pixel accuracy and mIoU on the validation set during training on the Pascal VOC dataset. 'Base' means a semantic segmentation network of PSPRes18 without a perceptual loss. 'Ours' represents the baseline network trained with a randomized perceptual loss.

Kernel size	mIoU (%)
1	70.71 ± 0.020
3	70.88 ± 0.025
5	70.93 ± 0.191
7	71.16 ± 0.122

Table 3 – The perceptual loss with different kernel sizes in the perceptual loss network with five layers. With a larger receptive field, the perceptual loss works slightly better, with increased computation in training.

Impact of multi-level losses. A few previous works pay attention to aggregate multi-level losses for the perceptual loss. We also conduct experiments to see if combining multi-level losses is helpful in our method. We employ the same baseline which achieves 69.60% of mIoU in semantic segmentation on Cityscapes and uses structure of '$1, 1, 1, 1, 1$' as the perceptual loss network. Adding the perceptual loss at the final layer of the random network alone can achieve 70.88% of mIoU. If we add five perceptual

loss from the output of each convolutional block with equal loss weight, it slightly harms the performance and achieve 70.73% of mIoU.

If we adjust the weights on the losses following [32] (*i.e.*, using scales of '$1/16$, $1/8$, $1/4$, $1/2$, 1' respectively), the result is improved slightly and achieves 70.93% of mIoU.

Combining multi-level of the losses may lead to slightly better accuracy with more hyper-parameters. Therefore, we only use the final layer as the generic perceptual loss in other tasks.

4.1.3 Initialization

In this section, we show the importance of initialization that we propose in Section 3.2. We initialize the perceptual loss network with the Gaussian distributions, the uniform distributions, the Xavier-normal initializer and our developed initialization methods in Section 3.2, and compare their results in . Table 4. We can see that experimental results are consistent with theoretical analysis.

Init. scheme	mIoU (%)
$\mathcal{N}(0, 1)$	$-$
$\mathcal{N}(0, 0.1)$	45.6
$\mathcal{N}(0, 0.01)$	68.5
$\mathcal{U}[-1, 1]$	$-$
$\mathcal{U}[-0.1, 0.1]$	51.7
$\mathcal{U}[-0.01, 0.01]$	69.2
Xavier-normal	69.8
Ours	**71.3**

Table 4 – Results with a few different initialization schemes. $\mathcal{N}(\mu, \sigma)$ represents the Gaussian distributions with mean of μ and a stander deviation of σ. $\mathcal{U}[a, b]$ represent a uniform distribution. $-$ means that the network fails to converge.

4.2. Dense Prediction Results

We show that the generic perceptual loss can work well in different tasks by taking the structured information into account during training. The perceptual loss network in this section refer to the VGG structure with 16 convolutional layers and 5 pooling layers, and the number of the input channel equals to the number of the task output channel. In semantic segmentation, the output channel equals to the number of class. In depth estimation and instance segmentation, the output channel is 1. The weight of the perceptual loss is set to 0.1.

4.2.1 Semantic Segmentation

Experiment settings. Experiments are conducted on three benchmarks, Cityscapes [4], Pascal VOC [5] and ADE20K [39]. On Cityscapes/Pascal VOC/ADE20K, the segmentation networks are trained by stochastic gradient descent (SGD) for 40K/20K/80K epochs with 8/16/16 training samples in the mini-batch, respectively. The learning rate is initialized as 0.01 and is multiplied by $(1 - \frac{\text{iter}}{\text{maxiter}})^{0.9}$. We randomly crop the images into 769×769, 512×512, 512×512 on these three datasets. Random scaling and random flipping are applied during training.

Experimental results. We employ three popular segmentation models with different model sizes, including a PSPNet [38] with ResNet18 as backbone (PSPRes18), a lightweight HRnet [30] with 18 layers (HRNetw18s) and the DeepLabV3+ [3] model with ResNet50 as the backbone. The corresponding soft targets are generated by the same architecture, but the backbone is replaced with ResNet101 or HRNet with 48 layers. The baseline models are trained with the cross-entropy loss, and we further add the perceptual loss initialized with random weights.

Table 6 reports the results. We can see that inclusion of the randomized perceptual loss can improve the performance over different datasets from 0.21% to 1.6%, and it also works with different architectures. If the unary term (pixel-wise loss) works sufficiently well, then additional pair-wise or other high-order loss becomes less useful.

4.2.2 Depth Estimation

Depth estimation is a typical per-pixel regression problem. We employ a plain ResNet50 as the backbone for depth estimation. The experiments are conducted on the NYUDV2 dataset [21]. The input images are cropped into the resolution of 385×385. The base learning rate is set to 0.0001. We train our model using SGD with a mini-batch size of 8 for 30 epochs. We employ a pixel-wise weighted cross-entropy [1] and the vitural normal loss [35].

Predictions are evaluated by the relative error. The results are shown in Table 7. Although the VNL have already considered the geometry information to some extent, the randomized perceptual loss network can still show further improvement.

4.2.3 Instance Segmentation

For instance segmentation, we apply the generic perceptual loss based on the open source framework AdelaiDet[2]. We employ the state-of-the-art method CondInst [26] as a strong baseline. ResNet50 is used as the backbone network for CondInst, and experiments are conducted on the MS COCO dataset.

Following [26], models are trained with SGD on 4 V100 GPUs for 90K iterations with the initial learning rate being 0.01 and a mini-batch of 8 images. Other training details the same as [26]. The learning rate is reduced by a factor of 10 at iteration 60K and 80K, respectively. Weight decay

[2] https://git.io/AdelaiDet

Method	Percep.	Box			Mask		
		AP	AP_{50}	AP_{75}	AP	AP_{50}	AP_{75}
CondInst [26]		36.91	55.29	39.94	33.42	53.00	35.56
CondInst [26]	✓	**37.44**	**55.69**	**40.51**	**33.69**	**53.33**	**35.82**

Table 5 – Results of the generic perceptual loss for instance segmentation. Both the detection results (Box AP (%)) and the segmentation results (Mask AP (%)) are improved by applying the generic perceptual loss.

Network	Param.	Percep.	mIoU
Cityscapes			
PSPRes18 [38]	22.9M		69.6%
PSPRes18 [38]	22.9M	✓	71.2% (↑1.6%)
HRNetw18s [32]	3.76M		73.61%
HRNetw18s [32]	3.76M	✓	74.17% (↑0.56%)
DeepLabV3+ [3]	39.3M		80.09 %
DeepLabV3+ [3]	39.3M	✓	80.70% (↑0.61%)
ADE20K			
PSPRes18 [38]	23.0M		33.8%
PSPRes18 [38]	23.0M	✓	34.2% (↑0.4%)
HRNetw18s [32]	3.79M		31.38%
HRNetw18s [32]	3.79M	✓	32.26% (↑0.88%)
DeepLabV3+ [3]	39.4M		42.72%
DeepLabV3+ [3]	39.4M	✓	42.95% (↑0.23%)
PascalVOC			
PSPRes18 [38]	22.9M		49.1%
PSPRes18 [38]	22.9M	✓	50.31% (↑1.21%)
HRNetw18s [32]	3.76M		65.20%
HRNetw18s [32]	3.76M	✓	65.41% (↑0.21%)
DeepLabV3+ [3]	39.3M		75.93%
DeepLabV3+ [3]	39.3M	✓	76.79% (↑0.86%)

Table 6 – Results of semantic segmentation on three datasets. ✓means that we employ the perceptual loss during training. The perceptual loss consistently improves the baseline across different datasets with different network structures.

Methods	WCE [1]	VNL [35]	Percep.	Rel. (%)
a	✓			14.5
b	✓		✓	14.0
c	✓	✓		13.6
d	✓	✓	✓	**13.2**

Table 7 – Depth estimation results. The generic perceptual loss complements the previous virtual normal loss [35].

and momentum are set to 0.0001 and 0.9, respectively. The experiment results are reported in Table 5. As bounding box detection is naturally a byproduct of CondInst, we compare results on both detection and instance segmentation tasks. With the generic perceptual loss, the performance of both tasks are improved on all metrics. This experiment again demonstrates that the proposed loss can benefit the *object detection* task, when a mask branch as in CondInst is added to an one-stage object detection method.

5. Conclusion

In this work, we have extended the widely used perceptual loss in image synthesis tasks to structured output learning tasks, and shows its usefulness. We argue that the randomly-weighted network can capture vital information among different spatial locations of the structural predictions. On a few image understanding tasks, including semantic segmentation, monocular depth estimation and instance segmentation, we demonstrate that the inclusion of this simple perceptual loss consistently improves accuracy. The proposed loss can be effortlessly applied to many dense prediction tasks in computer vision. The only cost is the extra computation overhead during training and inference complexity remains the same. Considering its simplicity and promising performance gain, we wish to see a wide application of this generic perceptual loss in computer vision.

References

[1] Yuanzhouhan Cao, Zifeng Wu, and Chunhua Shen. Estimating depth from monocular images as classification using deep fully convolutional residual networks. *IEEE Trans. Circuits Syst. Video Technol.*, 28(11):3174–3182, 2017.

[2] Hao Chen, Kunyang Sun, Zhi Tian, Chunhua Shen, Yongming Huang, and Youliang Yan. Blendmask: Top-down meets bottom-up for instance segmentation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 8573–8581, 2020.

[3] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proc. Eur. Conf. Comp. Vis.*, pages 801–818, 2018.

[4] 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 *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2016.

[5] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *Int. J. Comput. Vision*, 88(2):303–338, 2010.

[6] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 2002–2011, 2018.

[7] Adam Gaier and David Ha. Weight agnostic neural networks. In *Proc. Advances in Neural Inf. Process. Syst.*, pages 5364–5378, 2019.

[8] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2016.

[9] Kun He, Yan Wang, and John Hopcroft. A powerful generative model using random weights for the deep image representation. In *Proc. Advances in Neural Inf. Process. Syst.*, pages 631–639, 2016.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 1026–1034, 2015.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 770–778, 2016.

[12] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *Proc. Advances in Neural Inf. Process. Syst.*, pages 125–136, 2019.

[13] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei Efros. Image-to-image translation with conditional adversarial networks. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 1125–1134, 2017.

[14] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Proc. Eur. Conf. Comp. Vis.*, pages 694–711, 2016.

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet classification with deep convolutional neural networks. *Comm. of the ACM*, 60(6):84–90, 2017.

[16] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 4681–4690, 2017.

[17] Guosheng Lin, Chunhua Shen, Anton van den Hengel, and Ian Reid. Efficient piecewise training of deep structured models for semantic segmentation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 3194–3203, 2016.

[18] Yifan Liu, Ke Chen, Chris Liu, Zengchang Qin, Zhenbo Luo, and Jingdong Wang. Structured knowledge distillation for semantic segmentation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 2604–2613, 2019.

[19] Mihir Mongia, Kundan Kumar, Akram Erraqabi, and Yoshua Bengio. On random weights for texture generation in one layer CNNs. In *Proc. IEEE Int. Conf. Acous., Speech & Signal Process.*, pages 2207–2211, 2017.

[20] Reiichiro Nakano. A discussion of 'adversarial examples are not bugs, they are features': Adversarially robust neural style transfer. *Distill*, 4(8), 2019.

[21] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from RGBD images. In *Proc. Eur. Conf. Comp. Vis.*, 2012.

[22] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2018.

[23] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *Proc. Int. Conf. Learn. Representations*, 2015.

[24] Dong Su, Huan Zhang, Hongge Chen, Jinfeng Yi, Pin-Yu Chen, and Yupeng Gao. Is robustness the cost of accuracy?–a comprehensive study on the robustness of 18 deep image classification models. In *Proc. Eur. Conf. Comp. Vis.*, pages 631–648, 2018.

[25] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 1–9, 2015.

[26] Zhi Tian, Chunhua Shen, and Hao Chen. Conditional convolutions for instance segmentation. In *Proc. Eur. Conf. Comp. Vis.*, 2020.

[27] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. FCOS: Fully convolutional one-stage object detection. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 9627–9636, 2019.

[28] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2018.

[29] Hu Wang, Guansong Pang, Chunhua Shen, and Congbo Ma. Unsupervised representation learning by predicting random distances. *Proc. Int. Joint Conf. Artificial Intell.*, 2019.

[30] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2020.

[31] Lijun Wang, Jianming Zhang, Yifan Wang, Huchuan Lu, and Xiang Ruan. Cliffnet for monocular depth estimation with hierarchical embedding loss. In *Proc. Eur. Conf. Comp. Vis.*, pages 316–331, 2020.

[32] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 8798–8807, 2018.

[33] Xinlong Wang, Wei Yin, Tao Kong, Yuning Jiang, Lei Li, and Chunhua Shen. Task-aware monocular depth estimation for 3d object detection. In *Proc. AAAI Conf. Artificial Intell.*, volume 34, pages 12257–12264, 2020.

[34] Xinlong Wang, Rufeng Zhang, Tao Kong, Lei Li, and Chunhua Shen. Solov2: Dynamic, faster and stronger. *Proc. Advances in Neural Inf. Process. Syst.*, 2020.

[35] Yin Wei, Yifan Liu, Chunhua Shen, and Youliang Yan. Enforcing geometric constraints of virtual normal for depth prediction. *Proc. IEEE Int. Conf. Comp. Vis.*, 2019.

[36] Wei Yin, Jianming Zhang, Oliver Wang, Simon Niklaus, Long Mai, Simon Chen, and Chunhua Shen. Learning to recover 3d scene shape from a single image. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2021.

[37] Changqian Yu, Yifan Liu, Changxin Gao, Chunhua Shen, and Nong Sang. Representative graph neural network. *Proc. Eur. Conf. Comp. Vis.*, 2020.

[38] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 2881–2890, 2017.

[39] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2017.