Unveiling The Mask of Position-Information Pattern Through the Mist of Image Features

Chieh Hubert Lin [1] **Hung-Yu Tseng** [2] **Hsin-Ying Lee** [3] **Maneesh Kumar Singh** [4] **Ming-Hsuan Yang** [1 5 6]

Abstract

Recent studies have shown that paddings in convolutional neural networks encode absolute position information which can negatively affect the model performance for certain tasks. However, existing metrics for quantifying the strength of positional information remain unreliable and frequently lead to erroneous results. To address this issue, we propose novel metrics for measuring and visualizing the encoded positional information. We formally define the encoded information as Position-information Pattern from Padding (PPP) and conduct a series of experiments to study its properties as well as its formation. The proposed metrics measure the presence of positional information more reliably than the existing metrics based on PosENet and tests in F-Conv. We also demonstrate that for any extant (and proposed) padding schemes, PPP is primarily a learning artifact and is less dependent on the characteristics of the underlying padding schemes.

1. Introduction

Padding, one of the most fundamental components in neural network architectures, has received much less attention than other modules in the literature. In convolutional neural networks (CNNs), zero padding is frequently used perhaps due to its simplicity and low computational costs. This design preference remains almost unchanged in the past decade. Recent studies (Islam* et al., 2020; Islam et al., 2021b; Kayhan & Gemert, 2020; Innamorati et al., 2020) show that padding can implicitly provide a network model with positional information. Such positional information can cause unwanted side-effects by interfering and affecting other sources of position-sensitive cues (e.g., explicit coordinate inputs (Lin et al., 2022; Alsallakh et al., 2021a; Xu et al., 2021; Ntavelis et al., 2022; Choi et al., 2021), embeddings (Ge et al., 2022), or boundary conditions of the model (Innamorati et al., 2020; Alguacil et al., 2021; Islam et al., 2021a)). Furthermore, padding may lead to several unintended behaviors (Lin et al., 2022; Xu et al., 2021; Ntavelis et al., 2022; Choi et al., 2021), degrade model performance (Ge et al., 2022; Alguacil et al., 2021; Islam et al., 2021a), or sometimes create blind spots (Alsallakh et al., 2021a). Meanwhile, simply ignoring the padding pixels (known as no-padding or valid-padding) leads to the foveal effect (Alsallakh et al., 2021b; Luo et al., 2016) that causes a model to become less attentive to the features on the image border. These observations motivate us to thoroughly analyze the phenomenon of positional encoding including the effect of commonly used padding schemes.

Conducting such a study requires reliable metrics to detect the presence of positional information introduced by padding, and more importantly, quantify its strength consistently. We observe that the existing methods for detecting and quantifying the strength of positional information yield inconsistent results. In Section 3, we revisit two closely related evaluation methods, PosENet (Islam* et al., 2020) and F-Conv (Kayhan & Gemert, 2020). Our extensive experiments demonstrate that (a) metrics based on PosENet are unreliable with an unacceptably high variance, and (b) the Border Handling Variants (BHV) test in F-Conv suffers from unaware confounding variables in its design, leading to unreliable test results.

In addition, we observe all commonly-used padding schemes actually encode consistent patterns underneath the highly dynamic model features. However, such a pattern is rather obscure, noisy, and visually imperceptible for most paddings (except zeros-padding), which makes recognizing and analyzing it difficult. Fortunately, we show that such patterns can be consistently revealed with a sufficient number of samples by defining an optimal padding scheme (see Section 2.1 and Figure 1). We accordingly propose a new

*Equal contribution [1]University of California, Merced, USA [2]Meta Platforms, USA [3]Snap Inc., USA [4]Comcast, USA [5]Google Research, USA [6]Yonsei University, Korea. Correspondence to: Ming-Hsuan Yang <myang37@ucmerced.edu>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

The source codes and data collection scripts will be made publicly available: https://github.com/hubert0527/PPP.

evaluation paradigm and develop a method to consistently detect the presence of the Position-information Pattern from Padding (PPP), which is a persistent pattern embedded in the model features to retain positional information. We present two metrics to measure the response of PPP from the signal-to-noise perspective and demonstrate its robustness and low deviation among different settings, each with multiple trials of training.

To weaken the effect of PPP, in Section 2.4, we design a padding scheme with built-in stochasticity, making it difficult for the model to consistently construct such biases. However, our experiments show that the models can still circumvent the stochasticity and end up consistently constructing PPPs. These results suggest that a model likely constructs PPPs purposely to facilitate its training, rather than falsely or accidentally learning some filters that respond to padding features.

With reliable PPP metrics, we conduct a series of experiments to analyze the characteristics of PPP in Section 4.1 and to understand its correlation to the degradations caused by positional information in Section 4.3. Specifically, we analyze the formation of PPP throughout each model training process in Section 4.4. The results show PPPs are formed expeditiously at the early stage of model training, slowly but steadily strengthen through time, and eventually shaped in clear and complete patterns. These results show that a model intentionally develops and reinforces PPPs to facilitate its learning process. Moreover, we observe the PPPs of all pretrained networks are significantly stronger than those in their initial states. This indicates an unbiased training procedure is of great importance in resolving the critical failures caused by PPP in numerous vision tasks (Alsallakh et al., 2021a; Xu et al., 2021; Ge et al., 2022; Alguacil et al., 2021).

2. Observations and Methodology

In this section, we first define symbols for expressing the functionality of paddings and define the optimal-padding scheme. We then give a formal definition of Position-information Pattern from Padding (PPP) and utilize the optimal-padding scheme to develop propose a method to capture PPP and measure its response with two metrics.

2.1. Optimal Padding

The process of capturing an image from the real world can be simplified into two steps: (a) 3D information of the environment is first projected onto an infinitely large 2D plane, and then (b) the camera determines resolution as well as field-of-view to form a digital image from such infinitely large and continuous 2D signals (Liu et al., 2019; Ravi et al., 2020). Let $S^* = \{s^*_n\}_{n=1}^N$ be a collection of such infinitely large and continuous 2D signals, and the collection of 2D images captured by cameras at a spatial size (h_n, w_n) be $S' = \{s'_n\}_{n=1}^N$. Denote $(\Diamond)$ as the condition $0 < i < h_n$ and $0 < j < w_n$ both satisfied, where i and j are indexes of a pixel in the spatial dimension. A padding scheme can be used to generate a set of *algorithmically-padded* images $\hat{S} = \{\hat{s}_n\}_{n=1}^N$ by a padding function ρ:

$$\hat{s}_n[i,j] = \begin{cases} s'_n[i,j] = s^*[i,j] & \text{if } (\Diamond)\,, \\ \rho(s'_n, i, j) & \text{otherwise.} \end{cases} \tag{1}$$

We define a theoretical *optimally-padded* collection $S^\dagger = \{s^\dagger_n\}_{n=1}^N$ with an optimal-padding function $\rho^\dagger$ by:

$$s^\dagger_n[i,j] = \begin{cases} s'_n[i,j] & = s^*[i,j] \quad \text{if } (\Diamond)\,, \\ \rho^\dagger(s'_n, i, j) & = s^*[i,j] \quad \text{otherwise.} \end{cases} \tag{2}$$

This equation clarifies the optimal padding should consider the image collection procedure and seek the pixels presented in s^* back. In practice, without curated data, the *optimal*-padding scheme described in Eq. 2 is difficult to achieve. We describe how we relax this constraint in Section 2.3 and achieve equivalent quatities.

2.2. Positional-information Pattern from Padding

Despite the previous literature discovering the existence of positional information caused by the model paddings, there is still no clear definition for such information, and lacks effective metrics to detect or quantify it. Ideally, an effective metric for such positional information should have two properties. First, it is a spatial pattern, it contributes distinctive information to different spatial locations. Its shape enables the network to develop and exploit the absolute positional information of each pixel, eventually leading to the unattended and undesirable effects in certain tasks (Lin et al., 2022; Alsallakh et al., 2021a; Xu et al., 2021; Ntavelis et al., 2022; Choi et al., 2021; Ge et al., 2022; Alguacil et al., 2021). Second, as it represents the positional information purely contributed by the padding, it is a constant pattern irrelevant to the image contents. We accordingly name it the Positional-information Pattern from Padding (PPP).

Unfortunately, such a pattern shares space with image features, where the image features typically have very diverse appearances and high dimensionality. When these two signals *interfere* with each other, the appearance of PPP becomes extremely obscure and imperceptible in most cases (except zeros padding). Figure 1 shows if we visualize features sample-by-sample, there are no obvious differences between optimally-padded features (gray-scale surface) and algorithmically-padded features (colored surface). To address the issue, we show that, by assuming the interferences between PPP and image features to be random, its expectation over a large set of images will saturate to a constant bias and no longer hinder us from capturing PPP.

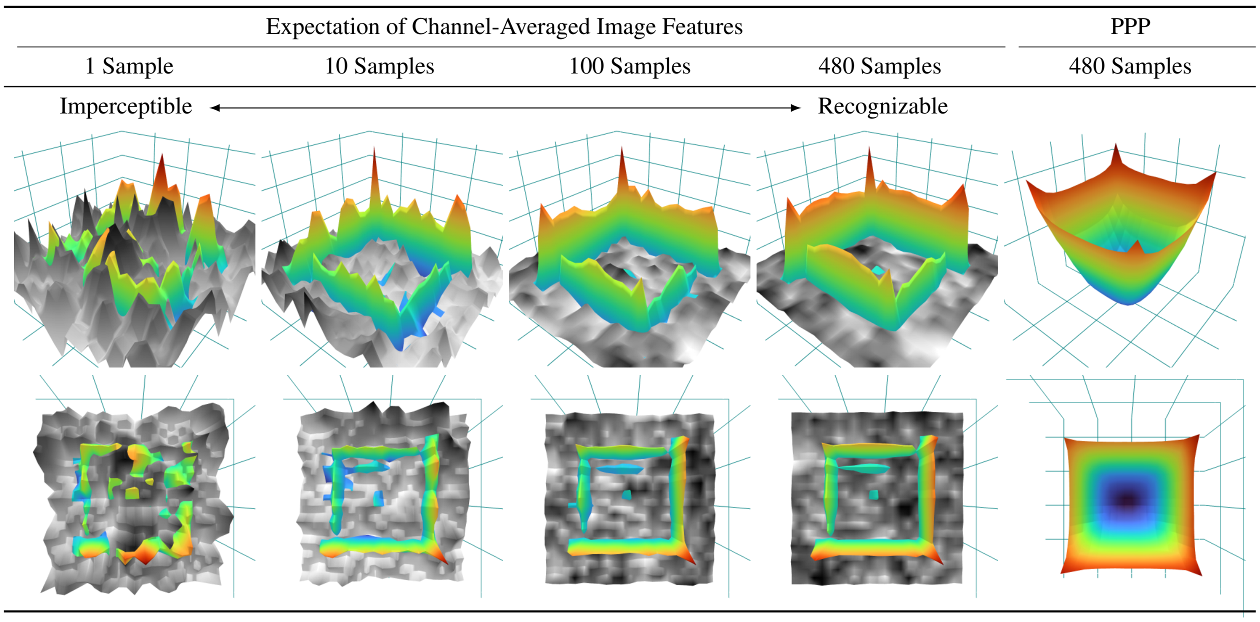


Figure 1. **Position-information Pattern from Padding (PPP).** We propose a method that can consistently and effectively extract PPPs through the distributional difference between optimally-padded (gray-scale surfaces) and algorithmically-padded features (colored surfaces). These feature surfaces are collected at a user-specified layer, and flattened into a 2D array by averaging the batch- and channel dimensions for visualization. Conceptually, a non-curated distribution (e.g., image content, the gray-scale distribution) will be averaged to a smooth distribution with sufficient samples. In contrast, a distribution curated/embedded with positional information will retain a shifted and static bias (i.e., positional information patterns) after averaging over a large set of samples. The results show that the two distributions become distinguishable as the number of samples increases. On the right-hand side, following the procedure in Section 2.2, we extract a clear view of PPP with the expectation of the pair-wise differences between optimally-padded and algorithmically-padded features. We render each visualization in a tilted view (first row) and a top view (second row). The colors represent the magnitude (blue/cold/weak to red/warm/strong) at each pixel. The features are extracted at the 3rd layer of interest (Appendix A) from a randn-padded (Section 2.4) ResNet50 pretrained on ImageNet.

Based on these observations and assumptions, we define PPP as the constant component independent of model inputs, and its presence is completely contributed by the existence of a padding scheme ρ. Given $\hat{S}$ and a model $F(\hat{s};\theta,\rho)$, which θ is the model parameters and ρ is a padding scheme applied to F. Let the model feature extracted at k-th layer be $f_{n,k} = F_k(\hat{s}_n;\theta,\rho)$, where F_k is the model from the first layer to the k-th layer. The PPP at k-th layer (PPP_k) can be formulated by:

$$\text{PPP}_k = \mathop{\mathbb{E}}_{n}\left[\ d\left(\ F_k(s_n^{\dagger};\theta,\rho^{\dagger})\ ,\ F_k(\hat{s}_n;\theta,\rho)\ \right)\ \right], \quad (3)$$

where $d(\cdot,\cdot)$ can be any distance function. We use ℓ_1 distance in this work, and accordingly, name the metric PPP-MAE.

Pitfalls: feature misalignment. It is important to note that, some CNN components can cause serious feature misalignment while computing PPP and leads to erroneous results. A typical example is *principal point shift*, where the uneven padding in stride-2 convolution causes the center of features slightly drifted, as shown in Appendix Figure 7. Since the measurement of PPP requires perfect alignment, such a drift should be carefully considered while integrating PPP into new architectures. We discuss the issue along with other pitfalls in Appendix A and provide three detailed examples of correcting the principal point shifting.

2.3. Simulated Optimal Padding

In practice, it is impossible to gain access to S^* for calculating the optimal padding $S^{\dagger}$ described in Eq. 2. But fortunately, given our goal in Eq. 3 is to analyze the model features within the (h_n, w_n) region, S^* is an overshoot of the data we actually required. Given a vision model $F(\hat{s};\theta,\rho)$ trained at a field-of-view (h_n, w_n) pixels, the receptive field of such vision model is (h_m, w_m) pixels (we show the computation in Appendix A), where $h_m \gg h_n$ and $w_m \gg w_n$. Let an alternative image collection $S^{\odot} = \{s_n^{\odot}\}_{n=1}^{N}$ at (h_m, w_m) pixels, the definition of receptive field implies $F_k(s_n^{\dagger};\theta,\rho)$ equals to $F_k(s_n^{\odot};\theta,\rho^{\dagger})$ for all k.

In other words, in terms of computing Eq. 3, $S^{\odot}$ is equivalent to S^* within the finite (h_n, w_n) region for a given model architecture. Therefore, we can simulate the procedure described in Eq. 1 and Eq. 2 using $S^{\odot}$ instead of $S^{\dagger}$, as long as $\forall s_n^{\odot} \in S^{\odot}$ the spatial size of $s_n^{\odot}$ is strictly larger than (h_m, w_m).

2.4. Randn Padding

Most of the existing padding schemes (e.g., zeros, reflect, replicate, circular) exhibit certain consistent patterns that can be easily detected by some designed convolutional kernels. One may argue that the nature of easy detectability can be a root cause of encouraging the models to learn to rely on these obvious patterns. This motivates us to design an additional sampling-based padding scheme without any consistent patterns, namely randn (i.e., random normal) padding, which produces dynamical values from a normal distribution while following the local statistics. We first determine the maximal and minimal values of a sliding window (which can be easily achieved with max-pooling), use the average of them as a proxy mean μ_p, and use the difference between the mean and the maximal value as a proxy standard deviation σ_p. For each padding location, we sample the padding value according to a normal distribution $\mathcal{N}(\mu_p, \sigma_p^2)$ from the nearest sliding window. We include more implementation details in Appendix A.

Aside from creating a pattern-less padding scheme with sampling, the design of randn padding is based on several factors. The sampled padding pixels are allowed to occasionally exceed the min/max bound of the sliding window. Without breaking the min/max bound can introduce detectable patterns in certain extreme cases, such as a gradient-like feature that has its maximal intensity at the top-left corner and minimal intensity at the bottom-right corner. We also design the padding scheme to follow the local distribution. The padding exhibits high entropy when the local variation is high, while degenerates to value repetition with imperceptible perturbations while padding a flat area. As such, not only do the padding pixels exhibit less pattern, but it also prevents the padding pixels from breaking the features in the border region. We later show that a model still deliberately and incredibly built up PPP over time even with such a sophisticated padding scheme.

3. Revisiting Prior Work

In this section, we first reproduce two experiments from the prior art, which aim to assess positional information from paddings. We show several critical design issues in these experiments and discuss how these problems affect the drawn conclusions. Finally, we propose two additional experiments to quantify the amount of positional information embedded in the paddings.

3.1. PosENet

Islam *et al.* show zeros-padding provides CNN models positional information cues, and propose PosENet (Islam* et al., 2020) to quantify the amount of positional information encoded within CNN features. A PosENet experiment involves several components: a pretrained CNN model F, a shallow CNN E_{pem} (i.e., position encoding module), an image dataset $X = \{x_i\}_{i=1}^N$ to examine, and a constant target pattern y (e.g., 2D Gaussian pattern). PosENet first extracts intermediate features at k-th layer with $f_{(i,k)} = F_k(x_i)$ using the pretrained CNN, and then optimizes E_{pem} to minimize $\mathbb{E}_{i,k}[||E_{pem}(f_{(i,k)}) - y||_2]$. Finally, the amount of positional information is quantified by the average Spearman's correlation (SPC) and Mean Absolute Error (MAE) overall $E_{pem}(f_{(i,k)})$ toward y.

A critical issue with PosENet is the use of an optimization-based metric. It is sensitive to hyperparameters with large variation. As shown in Table 2, for all the PosENet results, the standard deviation over five trials significantly dominates the differences between different types of paddings, and thus no definitive conclusions can be drawn. We also observed that PosENet can report NaN results in certain setups. Furthermore, PosENet quantifies the amount of positional information by the faithfulness of the final reconstruction. However, a better reconstruction does not have a clear relationship to *measuring* the strength and significance of positional information. For instance, PosENet sometimes shows responses to no-padding models, demonstrating it is a metric with an indefinite bias pending on the memorization ability of E_{pem}. Moreover, optimizing for pattern reconstruction is highly dependent on the underlying data distribution, simply changing the evaluation data distribution without changing the model weights can drastically change the PosENet numerical magnitudes and the conclusions of which model embeds the strongest positional information.

Another issue is that the no-padding scheme used in the E_{pem} module in PosENet is known to have the foveal effect (Alsallakh et al., 2021b; Luo et al., 2016), where a model pays less attention to the information on the edge of inputs. Using such a padding scheme for detecting positional information from paddings, which is mostly concentrated on the edge of the feature maps, is less effective. This is an inevitable dilemma as PosENet aims to identify positional information from the padding of the pretrained F, while applying any padding scheme to E_{pem} introduces intractable effects between the paddings of the two models.

3.2. F-Conv

Kayhan *et al.* propose a full-padding scheme (F-Conv) (Kayhan & Gemert, 2020) and demonstrate it is more translational invariant than the alternatives. One of the critical results is on "border handling variants" (Exp 2 of (Kayhan & Gemert, 2020)), which we call it BHV test. The BHV test creates a toy dataset, where each image has a black

Table 1. **Background color as a critical confounding variable in BHV test.** We show that using a grey background similar to Figure 2 leads to discrepant results. All paddings are using F-Conv (Kayhan & Gemert, 2020), which claims the similarity and dissimilarity tests should result in a similar performance. We additionally report an inconsistency rate as an even more sensitive metric. We mark the numbers that oppose the conclusions in (Kayhan & Gemert, 2020) with red. The standard deviations are reported among 10 individual trials. We report the full table in Appendix Table 5.

Bg Color	Padding	Similarity (%)	Dissimilar (%)	Inconsistency (%)
Black	Zeros	$89.24_{\pm 0.98}$	$89.24_{\pm 0.98}$	$18.02_{\pm 8.08}$
	Circular	$99.20_{\pm 0.23}$	$93.14_{\pm 2.88}$	$18.48_{\pm 3.55}$
	Reflect	$100.00_{\pm 0.00}$	$11.70_{\pm 15.38}$	$97.33_{\pm 6.16}$
Gray	Zeros	$100.00_{\pm 0.00}$	$4.77_{\pm 6.52}$	$96.79_{\pm 7.13}$
	Circular	$98.26_{\pm 0.50}$	$92.40_{\pm 4.23}$	$28.67_{\pm 6.18}$
	Reflect	$100.00_{\pm 0.00}$	$17.16_{\pm 12.19}$	$98.13_{\pm 3.44}$

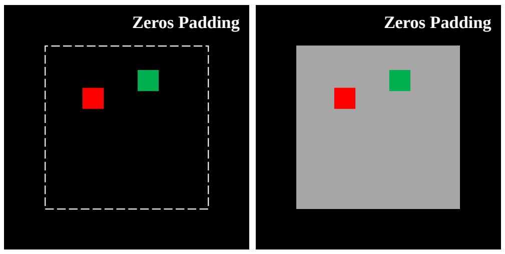


Figure 2. The BHV test trains a binary classifier to predict the relative position of the two colored squares. It hypothesizes if the padding provides no positional information, the classifier will only focus on the relative position of the two squares. (Left) The black background is a confounding variable. (Right) Zeros padding nolonger pads optimum values after changing the background color.

background with a green square and a red square in the foreground. The task is to predict if the red square is on the left of the green square (class 1), or vice versa (class 2). In addition, Kayhan *et al.* intentionally adds a *location bias* such that both squares are located in the upper half of the image for class 1, and located in the lower half of the image for class 2. During testing, a "similar test" inherits the same bias, while a "dissimilar test" exchanges the bias (i.e., both squares are in the lower half of the image for class 1). As a truly translation-invariant CNN model should not be affected by the location bias, it should focus on the relation between the red and green squares and perform similarly on both tests. Since the experimental results show that F-Conv performs best on the dissimilar test, it is concluded that F-Conv is less sensitive to the location bias. The authors also conclude the circular padding performs worse due to the behavior of wrapping the pixels to the other side of the image, which leads to confusion between two classes.

However, as shown in Figure 2, we find the experimental design does not consider a crucial confounding variable: the black background has a zero intensity, making zeros padding the optimal padding that perfectly follows the background distribution. In Table 1, we show that the dissimilar test is no longer in favor of F-Conv zeros after changing the background color to grey. We also show that F-Conv replicate and F-Conv circular perform best on the dissimilar test, which is different from the original observation.

Finally, we report an additional inconsistency rate to show that the CNN architecture used in the BHV test actually has access to the absolute position of the squares. Given a random sample in class 1, we create a *trajectory* of samples by simultaneously moving the two squares to the bottom of the canvas and recording the CNN-model prediction in all intermediate states. We label a trajectory to be *inconsistent* if the prediction of the CNN-model switches classes at any step of the trajectory. A CNN model with no access to the absolute-position information should have all trajectories maintaining consistent predictions, with 0% inconsistency. Table 1 shows the inconsistent ratio over 228 uniformly sampled trajectories, where all models maintain high inconsistency rates, even with a no-padding architecture. These results show that the CNN model used in the BHV test is not translation invariant. This can be attributed to that a CNN model has a large receptive field covering the whole experiment canvas, therefore capable of gradually constructing absolute coordinates for each input pixel. Note that we only show the design of the BHV test is not suitable for quantifying the amount of positional information exhibited in a CNN model. Such a conclusion does not imply that F-Conv cannot potentially improve the translation-invariant property of CNNs.

4. Experiments and Analysis

Datasets Since most vision models are trained on tasks for recognizing objects, an image collection containing a diverse object appearance is more suitable for the task. As mentioned in Section 2.3, evaluating PPP requires images at a large field-of-view, in practice, we collect three image datasets at $2{,}048^2$ pixels, which is larger than the receptive field of all the models we tested. The three datasets at $2{,}048^2$ pixels are (a) 480 satellite images crawled from Google Map, (b) 1,024 images synthesized by InfinityGAN (Lin et al., 2022) trained with Flickr-Landscape dataset, and (c) 1,024 images synthesized by InfinityGAN trained with LSUN-Tower (Yu et al., 2015) dataset. In addition, we also evaluate PPP on three computer vision datasets: (d) ImageNet (Deng et al., 2009) validation split, (e) MS-COCO (Lin et al., 2014), (f) PASCAL-S (Li et al., 2014) used in PosENet. For (d) and (e), we filter and only keep images with a resolution larger than 512^2 to avoid an unreasonable image resize ratio, then all images in the (d-e) settings are resized to the receptive field based on the tested model architecture. While evaluating PPP, we crop the input images depending on the receptive field and principal point shifts from each model (see Appendix A for details). We will release the script for collecting and composing these large images.

4.1. Visualizing Position-information Pattern from Padding (PPP)

We start with visualizing PPP in Figure 3. All the visualizations are conducted at the 3rd layer of interest as detailed in Appendix A. We compute PPP using Eq. 3 and ℓ_1 norm as the distance metric, then average the resulting PPP in the channel dimension to generate a gray-scale image. Since the quantities are small and difficult to perceive, we normalize the gray-scale image to $[0, 1]$ range, and thus the colors between images are not directly comparable.

In all scenarios, PPP noticeably spreads out after being pretrained on ImageNet. In Table 4, the PPP-MAE of the VGG19 and ResNet50 also reflects that the response of PPP is significantly strengthened after model training. That is, the model training has substantial effects on the construction of PPP. Although the formation of padding pattern is suggested to be mainly caused by the distributional difference between features and paddings (Alsallakh et al., 2021a), our results show that it only increases the response slightly, compared to the considerable PPP-MAE gain through training.

Another intriguing observation is that, despite some variations in the detailed patterns, the overall structure of PPP remains similar. Regardless of padding minimum values with zero-padding (consider the features are processed with ReLU activation), randn-padding that can sometimes produce large quantities by chance, or the unbalanced initial state of ResNet50 caused by strided convolution (the first row of ResNet50 in Figure 3), all models tend to have the maximal PPP response in the corner of the features after fully trained. While the underlying mechanism causing such consistent preferences remains unknown, such preferences may be an important factor to consider in future model design.

4.2. Quantifying PPP and Comparing with PosENet

Table 2 shows the measurements of PPP and PosENet on various architectures and padding schemes. We train five models for each setup and measure the standard deviation of these models. Our PPP-MAE has significantly lower standard deviations compared to PosENet, where the standard deviation of PosENet dominates the differences between padding variants, and thus the quantities from PosENet cannot provide sufficient information for any analysis. Evaluating the true mean of PosENet requires an even larger number of pretrained models, each requiring full training on the target dataset (e.g., ImageNet), which is impractical in reality. The main reason that PosENet has such a large variation is due to its optimization-based formulation, and thus the final quantities highly depend on the convergence of the PosENet training. In fact, we also observe a similar level of standard deviation even when the PosENet is measured on the same model for multiple trials. On the other hand, PPP is based on a closed-form formulation, and thus the variations are only introduced by the differences among the parameters of the pretrained models. Furthermore, PosENet often reports positive SPC responses from no-padding models, as shown in its large standard deviation. In contrast, PPP has zero response to no-padding models by definition, and therefore is less biased for measuring the positional information from padding.

Although certain paddings seem to have slightly lower PPP-

Table 2. **Comparing PosENet and our PPP metric. Most of the PosENet results are indistinguishable due to high variation.** We show a subset of results with VGG-19, the complete table is reported in Appendix Table 6, 7 and 8. The standard deviation is computed over five different pretrained models. We report MAE metric for both PosENet and our PPP, use 2D Gaussian as PosENet reconstruction pattern, and measure PPP-MAE at the 4th layer of interest. (↑) indicates a higher value corresponds to stronger positional information (vice versa for (↓)). For each group of pretrained models, we label the strongest positional information response with red, and the experiments within its standard deviation range with blue. A good metric should have red entries concentrated under a single padding scheme and a few blue entries.

Padding	Eval Dataset	PosENet-MAE (↓)	PPP-MAE (↑)	Accuracy (%)
Zeros	GMap	$0.196_{\pm 0.006}$	$0.0176_{\pm 0.0005}$	$74.0972_{\pm 0.0870}$
	InfGAN-flickr	$0.183_{\pm 0.007}$	$0.0163_{\pm 0.0006}$	
	InfGAN-tower	$0.173_{\pm 0.010}$	$0.0179_{\pm 0.0001}$	
	ImageNet-val	$0.237_{\pm 0.178}$	$0.0164_{\pm 0.0003}$	
	MS-COCO	$0.200_{\pm 0.173}$	$0.0173_{\pm 0.0002}$	
	PASCAL-S	$0.081_{\pm 0.145}$	$0.0163_{\pm 0.0002}$	
Circular	GMap	$0.197_{\pm 0.007}$	$0.0158_{\pm 0.0006}$	$74.4716_{\pm 0.0863}$
	InfGAN-flickr	$0.185_{\pm 0.009}$	$0.0137_{\pm 0.0004}$	
	InfGAN-tower	$0.176_{\pm 0.009}$	$0.0184_{\pm 0.0005}$	
	ImageNet-val	$0.175_{\pm 0.174}$	$0.0161_{\pm 0.0003}$	
	MS-COCO	$0.148_{\pm 0.165}$	$0.0167_{\pm 0.0003}$	
	PASCAL-S	$0.083_{\pm 0.164}$	$0.0154_{\pm 0.0003}$	
Reflect	GMap	$0.196_{\pm 0.007}$	$0.0158_{\pm 0.0002}$	$74.0516_{\pm 0.0621}$
	InfGAN-flickr	$0.185_{\pm 0.008}$	$0.0146_{\pm 0.0008}$	
	InfGAN-tower	$0.177_{\pm 0.009}$	$0.0168_{\pm 0.0005}$	
	ImageNet-val	$0.183_{\pm 0.193}$	$0.0157_{\pm 0.0004}$	
	MS-COCO	$0.173_{\pm 0.170}$	$0.0165_{\pm 0.0002}$	
	PASCAL-S	$0.102_{\pm 0.182}$	$0.0153_{\pm 0.0003}$	
Replicate	GMap	$0.197_{\pm 0.006}$	$0.0144_{\pm 0.0009}$	$73.9964_{\pm 0.1079}$
	InfGAN-flickr	$0.184_{\pm 0.007}$	$0.0128_{\pm 0.0012}$	
	InfGAN-tower	$0.173_{\pm 0.010}$	$0.0156_{\pm 0.0006}$	
	ImageNet-val	$0.229_{\pm 0.181}$	$0.0143_{\pm 0.0006}$	
	MS-COCO	$0.209_{\pm 0.169}$	$0.0149_{\pm 0.0006}$	
	PASCAL-S	$0.110_{\pm 0.176}$	$0.0139_{\pm 0.0004}$	
Randn	GMap	$0.195_{\pm 0.006}$	$0.0182_{\pm 0.0012}$	$73.7716_{\pm 0.0758}$
	InfGAN-flickr	$0.185_{\pm 0.007}$	$0.0167_{\pm 0.0008}$	
	InfGAN-tower	$0.181_{\pm 0.010}$	$0.0186_{\pm 0.0012}$	
	ImageNet-val	$0.204_{\pm 0.188}$	$0.0173_{\pm 0.0008}$	
	MS-COCO	$0.153_{\pm 0.180}$	$0.0182_{\pm 0.0008}$	
	PASCAL-S	$0.099_{\pm 0.201}$	$0.0166_{\pm 0.0008}$	
NoPad	GMap	$0.204_{\pm 0.013}$	$0.0000_{\pm 0.0000}$	$62.0396_{\pm 0.0830}$
	InfGAN-flickr	$0.187_{\pm 0.012}$	$0.0000_{\pm 0.0000}$	
	InfGAN-tower	$0.172_{\pm 0.014}$	$0.0000_{\pm 0.0000}$	
	ImageNet-val	$0.048_{\pm 0.241}$	$0.0000_{\pm 0.0000}$	
	MS-COCO	$0.031_{\pm 0.231}$	$0.0000_{\pm 0.0000}$	
	PASCAL-S	$0.033_{\pm 0.257}$	$0.0000_{\pm 0.0000}$	

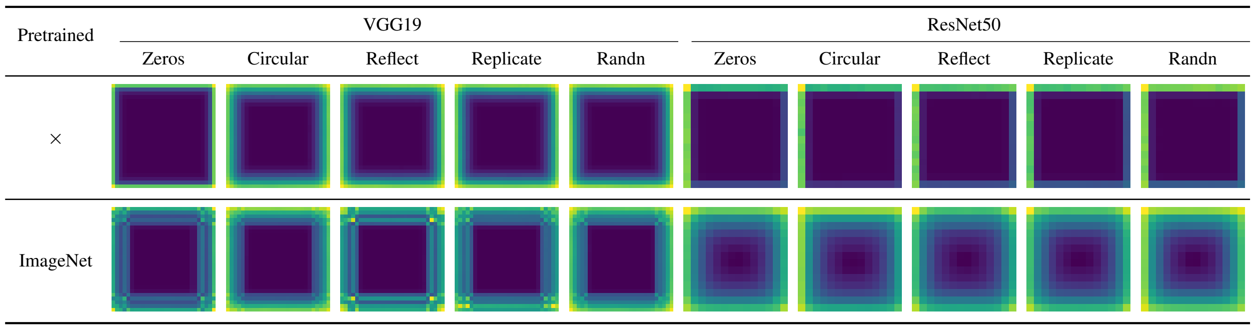


Figure 3. **Visualization of Position-Information Pattern from Padding (PPP).** The visualizations are calculated based on Eq. 3 over 480 GMap samples extracted at the 3rd layer-of-interest (Appendix A). The results show that the pretrained model significantly reinforces PPP compared to randomly initialized networks. Note that each image is normalized to $[0, 1]$ separately, therefore the colors between images are not comparable. More visualizations are presented in Appendix E.

MAE than other paddings, in Table 4, we find the differences are not significant when comparing the extremely low PPP-MAE from most of the randomly initialized networks. In most cases, the network can effectively construct its PPP, even with the highly stochastic randn padding. The only exception seems to be the case of randn padding in the salient object detection (SOD) task, where the network fails to achieve a compatible performance with other paddings[1]. The results show that the model training plays an important role in the formation of PPP, and perhaps its contribution is much larger than which underlying padding scheme is being used. This motivates us to further analyze the PPP formulation during model training.

4.3. Correlation with Generalization

Despite a sufficiently low standard deviation being a critical requirement for a usable metric, it is still unclear if our proposed PPP metric can be used to measure the generalization issues caused by the positional information patterns. Therefore, we design an additional experiment to verify the correlation between the positional information metrics (i.e., PPP and PosENet) and the generalization gaps.

Evaluating the degradation. However, for most computer vision tasks, it is not straightforward to recognize which degradation is purely caused by the positional information. We found the semantic image synthesis problem is an ideal testbed for such a problem, where the goal of the task is to synthesize a realistic image based on a semantic segmentation map as the conditional input. The task is an ideal choice as its evaluation does not require labels, therefore it is easier to obtain and evaluate on test data at different field-of-views (not resolution). We use SPADE (Park et al., 2019) in this case study.

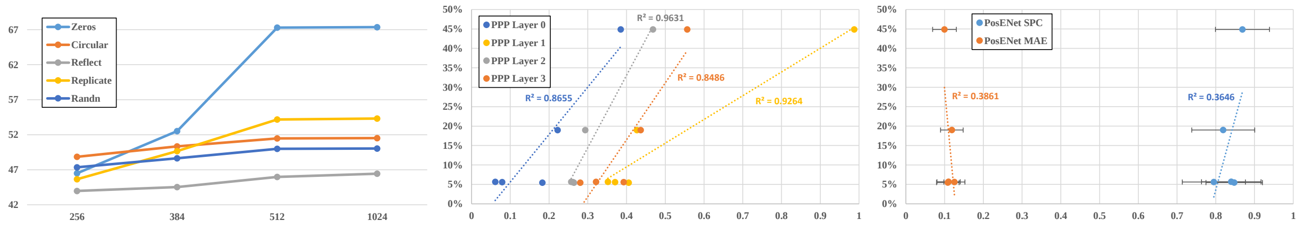
[1]We use the same setting as PosENet to evaluates PiCANet (Liu et al., 2018) on the SOD task. PiCANet is initialized by a model pretrained on ImageNet (with zero padding). The discrepancy in the padding scheme can be the major cause of failure while training the network on SOD task with randn padding.

Table 3. **CropFID measures degradation due to positional information.** We evaluate CropFID on SPADE models trained on the Flickr Landscapes dataset at 256^2 field-of-view, and tested at various field-of-views. We report the degradation percentage in parentheses, and mark the degraded cases in red.

Test Size	Zeros	Circular	Reflect	Replicate	Randn
256^2	46.49 (0%)	48.85 (0%)	43.97 (0%)	45.64 (0%)	47.35 (0%)
384^2	52.52 (**+13%**)	50.33 (**+3%**)	44.53 (**+1%**)	49.67 (**+9%**)	48.64 (**+3%**)
512^2	67.30 (**+45%**)	51.47 (**+5%**)	45.97 (**+5%**)	54.17 (**+19%**)	49.98 (**+6%**)
1024^2	67.36 (**+45%**)	51.52 (**+5%**)	46.43 (**+6%**)	54.31 (**+19%**)	50.03 (**+6%**)

Similar to image recognition tasks, semantic image synthesis models also learn to exploit the positional information pattern and synthesize contents based on the location of the pixel. After the model is trained at a certain field-of-view (e.g., 256^2 pixels), it is adapted to the specific positional information pattern at such a field-of-view. Consequently, these models will suffer from performance degradation if tested at different field-of-views (e.g., 1024^2 pixels, four times field-of-view at the same resolution), due to the distorted positional information patterns after changing the field-of-view. We show a few samples of such degradation in Appendix Figure 8.

Measuring degradation with CropFID. We measure such degradation with CropFID, where we always center-crop the synthesized image to a certain field-of-view (e.g., 256^2 pixels, again) regardless of the current input condition field-of-view, then measure the FID (Heusel et al., 2017) between the cropped synthetic images with real images. Since the CropFID only evaluates the center region of the image patch, the additional field-of-view that appeared at testing will not be evaluated. Therefore, the degradation of the network performance is purely caused by the discrepancy of the positional information in the center region of the image. By separately measuring CropFID at different field-of-views,



(a) CropFID (b) Degradation-PPP Correlation (c) Degradation-PosENet Correlation

Figure 4. **PPP has a stronger correlation to the degradation caused by positional information.** In (a), we first show that a SPADE (Park et al., 2019) model trained at 256^2 pixels has degraded CropFID performance in all larger field-of-view settings in all types of paddings. Then, in (b), we show PPP (x-axis) has a strong correlation to such a degradation (y-axis). Meanwhile, in (c), PosENet (Islam* et al., 2020) (x-axis) has a weaker correlation to such degradation, along with a very high standard deviation.

Table 4. **Significant PPP gain from model training.** We measure PPP-MAE on GMap with randomly initialized and fully trained models. The results show a consistent and significant increment of PPP is developed after the model is fully trained.

Model	Pretrained	Padding				
		Zeros	Circular	Reflect	Replicate	Randn
VGG-19	×	0.0132	0.0000	0.0000	0.0000	0.0000
	ImageNet	0.0176	0.0158	0.0158	0.0144	0.0182
ResNet50	×	0.0052	0.0032	0.0018	0.0015	0.0020
	ImageNet	0.0162	0.0188	0.0150	0.0150	0.0147

we can accurately measure the amount of degradation purely caused by the change of positional information pattern.

Experiment setup. For the dataset, we use the Flickr-landscape dataset from InfinityGAN (Lin et al., 2022), where all images are at 1024^2 pixels. Following the procedure described in SPADE (Park et al., 2019), we use UperNet101 (Zhou et al., 2017) to automatically label segmentation maps for all images. We use 4,800 test images to evaluate all the metrics (i.e., CropFID, PPP, and PosENet). To ensure the visual representations learned by the network do not have a large train-test domain gap, the image content should maintain a similar resolution during both the training and testing phases. Therefore, we center-crop (instead of resize) images to 256^2 for training, and evaluate CropFID at 256^2.

Observations. In Figure 4a and Table 3, we first show the existence and the severeness of the degradation while changing the field-of-view to different levels at testing. Not only the degradation consistently appears in all types of padding schemes, but the degradation can be up to 50% of the original CropFID for certain padding schemes.

In Figure 4b and 4c, we show that PPP has a stronger correlation to the degradation caused by changing the image field-of-view. We also report the coefficient of determination (R^2), where the R^2 in all layers are typically larger than 0.8 for PPP, while lower than 0.4 for PosENet, showing PPP has a stronger correlation to the degradation caused by the positional information.

4.4. Chronological PPP

To understand the formulation of PPP through time, we snapshot checkpoints every 10 epochs for all training episodes. By measuring the PPP-MAE at all the checkpoints, we plot a chronological curve and monitor the progress of PPP. We train 5 individual models for each pair of model-padding setting and report the standard deviations, which demonstrates the significance of the trend.

Figure 5 shows all models achieve a significant gain of PPP within the first 10 epochs in all intermediate layers. Most models continuously increase their PPP as training proceeds, especially in the fourth layer of interest, which is the last output from the convolutional layers before the final linear projection. Another interesting observation is that our randn padding, which is designed to be less easily detectable with built-in stochasticity, indeed shows less PPP built-up at the intermediate stages in certain layers. However, the network still adjusts the behavior and ends up forming complete PPPs at the fourth layer of interest in all scenarios. All these shreds of evidence show that the network builds PPP purposely as a favorable representation to assist its learning.

5. Conclusion and Limitations

In this paper, we develop a reliable method for measuring PPP and conduct a series of analyses toward understanding the formation and properties of PPP. Through a large-scale study, we demonstrate that PPP is a representation that the network favorably develops as a part of its learning process, and its formation has weak connections to the underlying padding algorithm. We show that reliable PPP metrics are important steps for understanding the effects of PPPs in different tasks, and useful for measuring the effectiveness of future methods in debiasing PPP.

However, an unfortunate and inevitable limitation of the

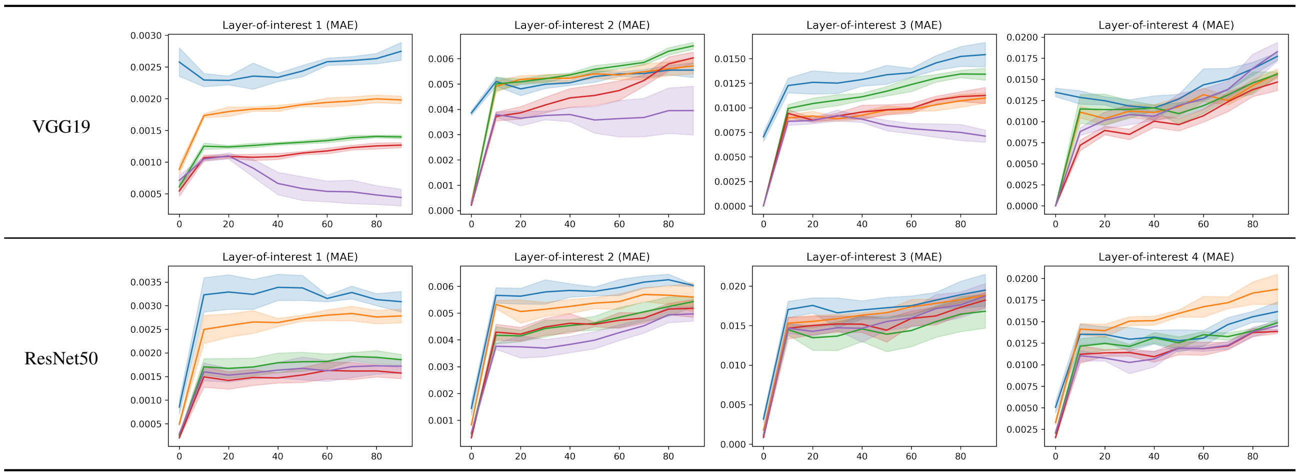


Figure 5. **Chronological PPP.** We quantify PPP every 10 epochs and plot its development in four different layer of depth (the rightmost layer is the one closest to model output). All curves consistently show a sudden surge at the early stage, and all the later layers are slowly but steadily gaining stronger PPP until the end of training. The shadow region represents standard deviations among 5 individual training episodes. The colors represent zeros, circular, reflect, replicate, and randn paddings.

PPP metrics is that their measure is biased by the model architecture and parameters. Since the PPP metrics are based on the distributional differences between the paired model outputs (i.e., optimal padding to algorithmic padding), different architecture and layers of depth exhibit different and intractable biases due to different interactions between PPP and model parameters. Such a bias makes PPP metrics less comparable while dissecting models with different architectures or parameter distributions (e.g., weight decay and weight normalization), which is important for studying the effect of architectural changes. However, this limitation is inevitable for any (and all existing) metric that attempts to measure PPP using the outputs of a model. We note future studies in measuring PPP without model inferences will be an important step toward tackling and understanding the property of PPP under different architectural choices.

6. Acknowledgements

We sincerely thank the discussions and feedback from Tsun-Hsuan Wang that help to develop the concept of this paper. This work is supported in part by a gift from Verisk Analytics Inc.

Our implementation reuses codes from several open-source codebases, which greatly supports the development of our project. Sincerely thanks to the owners/developers of F-Conv (Oskyhn, 2019), PyTorch (Paszke et al., 2019), torchvision (Pytorch, 2016), and Pytorch-cifar (Kuangliu, 2017).

References

Alguacil, A., Pinto, W. G., Bauerheim, M., Jacob, M. C., and Moreau, S. Effects of boundary conditions in fully convolutional networks for learning spatio-temporal dynamics. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2021. 1, 2

Alsallakh, B., Kokhlikyan, N., Miglani, V., Yuan, J., and Reblitz-Richardson, O. Mind the pad – {cnn}s can develop blind spots. In *International Conference on Learning Representations*, 2021a. 1, 2, 6

Alsallakh, B., Miglani, V., Kokhlikyan, N., Adkins, D., and Reblitz-Richardson, O. Are convolutional networks inherently foveated? In *SVRHM 2021 Workshop at NeurIPS*, 2021b. 1, 4

Choi, J., Lee, J., Jeong, Y., and Yoon, S. Toward spatially unbiased generative models. In *IEEE International Conference on Computer Vision*, 2021. 1, 2

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009. 5

Ge, S., Hayes, T., Yang, H., Yin, X., Pang, G., Jacobs, D., Huang, J.-B., and Parikh, D. Long video generation with time-agnostic vqgan and time-sensitive transformer. *arXiv preprint arXiv:2204.03638*, 2022. 1, 2

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *Neural Information Processing Systems*, 2017. 7

Innamorati, C., Ritschel, T., Weyrich, T., and Mitra, N. J. Learning on the edge: Investigating boundary filters in cnns. *International Journal of Computer Vision*, 2020. 1

Islam*, M. A., Jia*, S., and Bruce, N. D. B. How much position information do convolutional neural networks encode? In *International Conference on Learning Representations*, 2020. 1, 4, 8

Islam, M. A., Kowal, M., Jia, S., Derpanis, K. G., and Bruce, N. Boundary effects in {cnn}s: Feature or bug? `https://openreview.net/forum?id=M4qXqdw3xC`, 2021a. 1

Islam, M. A., Kowal, M., Jia, S., Derpanis, K. G., and Bruce, N. D. Position, padding and predictions: A deeper look at position information in cnns. *arXiv preprint arXiv:2101.12322*, 2021b. 1

Kayhan, O. S. and Gemert, J. C. v. On translation invariance in cnns: Convolutional layers can exploit absolute spatial location. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 1, 4, 5

Kuangliu. pytorch-cifar. `https://github.com/kuangliu/pytorch-cifar`, 2017. 9

Li, Y., Hou, X., Koch, C., Rehg, J. M., and Yuille, A. L. The secrets of salient object segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014. 5

Lin, C. H., Cheng, Y.-C., Lee, H.-Y., Tulyakov, S., and Yang, M.-H. InfinityGAN: Towards infinite-pixel image synthesis. In *International Conference on Learning Representations*, 2022. 1, 2, 5, 8

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, 2014. 5

Liu, N., Han, J., and Yang, M.-H. Picanet: Learning pixel-wise contextual attention for saliency detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 7

Liu, S., Li, T., Chen, W., and Li, H. Soft rasterizer: A differentiable renderer for image-based 3d reasoning. In *IEEE International Conference on Computer Vision*, 2019. 2

Luo, W., Li, Y., Urtasun, R., and Zemel, R. Understanding the effective receptive field in deep convolutional neural networks. In *Neural Information Processing Systems*, 2016. 1, 4

Ntavelis, E., Shahbazi, M., Kastanis, I., Timofte, R., Danelljan, M., and Van Gool, L. Arbitrary-scale image synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 1, 2

Oskyhn. Cnns-without-borders. `https://github.com/oskyhn/CNNs-Without-Borders`, 2019. 9

Park, T., Liu, M.-Y., Wang, T.-C., and Zhu, J.-Y. Semantic image synthesis with spatially-adaptive normalization. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 7, 8

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. In *Neural Information Processing Systems*, 2019. 9

Pytorch. vision. `https://github.com/pytorch/vision`, 2016. 9

Ravi, N., Reizenstein, J., Novotny, D., Gordon, T., Lo, W.-Y., Johnson, J., and Gkioxari, G. Accelerating 3d deep learning with pytorch3d. *arXiv preprint arXiv:2007.08501*, 2020. 2

Xu, R., Wang, X., Chen, K., Zhou, B., and Loy, C. C. Positional encoding as spatial inductive bias in gans. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 1, 2

Yu, F., Seff, A., Zhang, Y., Song, S., Funkhouser, T., and Xiao, J. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015. 5

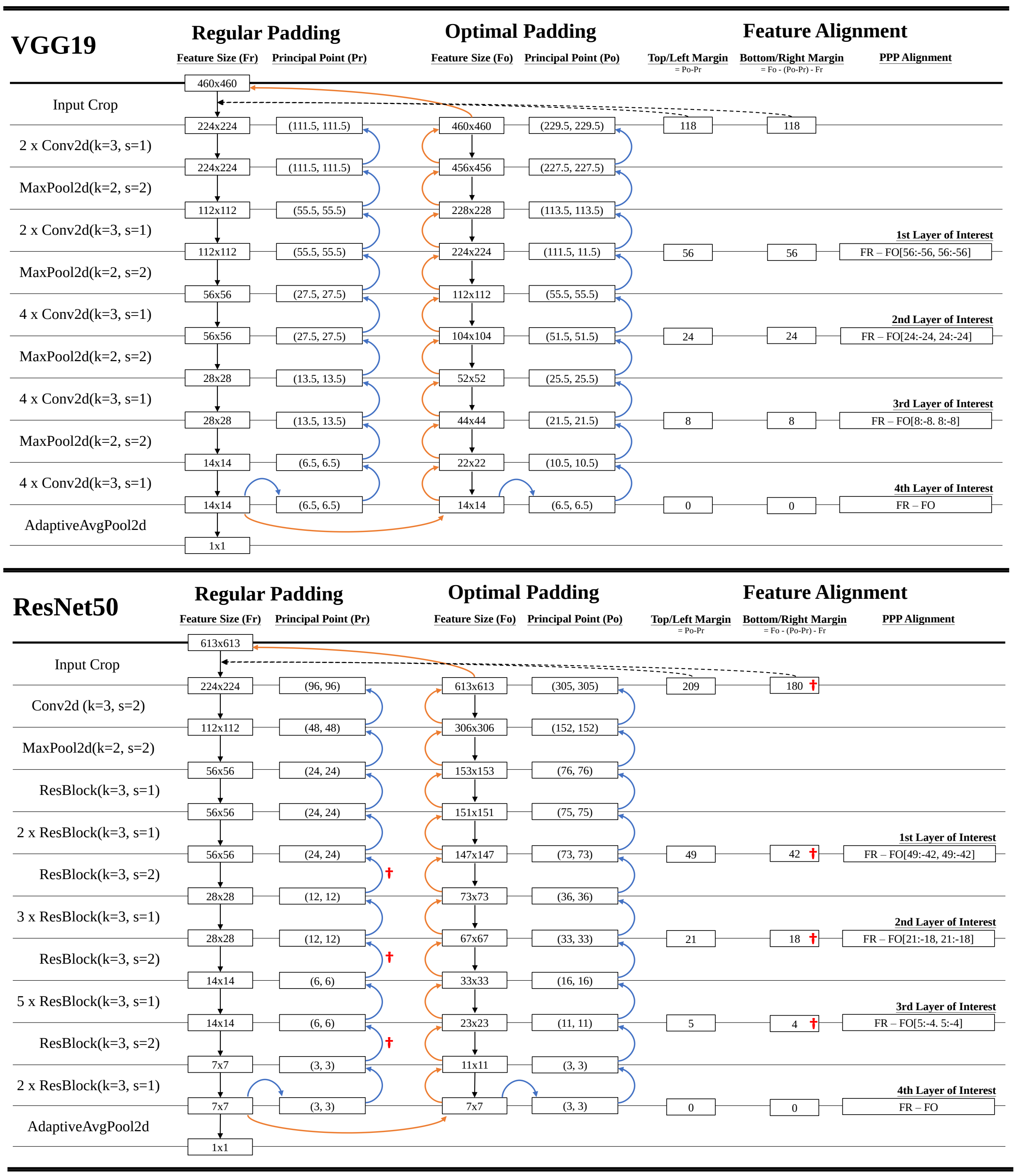
Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 12

Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., and Torralba, A. Scene parsing through ade20k dataset. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 8

Supplementary Material

A. Implementation Details

A.1. Architecture and Feature Alignments



VGG19

	Regular Padding		Optimal Padding		Feature Alignment		
	Feature Size (Fr)	Principal Point (Pr)	Feature Size (Fo)	Principal Point (Po)	Top/Left Margin = Po-Pr	Bottom/Right Margin = Fo - (Po-Pr) - Fr	PPP Alignment
	460x460						
Input Crop							
	224x224	(111.5, 111.5)	460x460	(229.5, 229.5)	118	118	
2 x Conv2d(k=3, s=1)							
	224x224	(111.5, 111.5)	456x456	(227.5, 227.5)			
MaxPool2d(k=2, s=2)							
	112x112	(55.5, 55.5)	228x228	(113.5, 113.5)			
2 x Conv2d(k=3, s=1)							
	112x112	(55.5, 55.5)	224x224	(111.5, 11.5)	56	56	1st Layer of Interest FR – FO[56:-56, 56:-56]
MaxPool2d(k=2, s=2)							
	56x56	(27.5, 27.5)	112x112	(55.5, 55.5)			
4 x Conv2d(k=3, s=1)							
	56x56	(27.5, 27.5)	104x104	(51.5, 51.5)	24	24	2nd Layer of Interest FR – FO[24:-24, 24:-24]
MaxPool2d(k=2, s=2)							
	28x28	(13.5, 13.5)	52x52	(25.5, 25.5)			
4 x Conv2d(k=3, s=1)							
	28x28	(13.5, 13.5)	44x44	(21.5, 21.5)	8	8	3rd Layer of Interest FR – FO[8:-8. 8:-8]
MaxPool2d(k=2, s=2)							
	14x14	(6.5, 6.5)	22x22	(10.5, 10.5)			
4 x Conv2d(k=3, s=1)							
	14x14	(6.5, 6.5)	14x14	(6.5, 6.5)	0	0	4th Layer of Interest FR – FO
AdaptiveAvgPool2d							
	1x1						

ResNet50

	Regular Padding		Optimal Padding		Feature Alignment		
	Feature Size (Fr)	Principal Point (Pr)	Feature Size (Fo)	Principal Point (Po)	Top/Left Margin = Po-Pr	Bottom/Right Margin = Fo - (Po-Pr) - Fr	PPP Alignment
	613x613						
Input Crop							
	224x224	(96, 96)	613x613	(305, 305)	209	180 †	
Conv2d (k=3, s=2)							
	112x112	(48, 48)	306x306	(152, 152)			
MaxPool2d(k=2, s=2)							
	56x56	(24, 24)	153x153	(76, 76)			
ResBlock(k=3, s=1)							
	56x56	(24, 24)	151x151	(75, 75)			
2 x ResBlock(k=3, s=1)							
	56x56	(24, 24)	147x147	(73, 73)	49	42 †	1st Layer of Interest FR – FO[49:-42, 49:-42]
ResBlock(k=3, s=2)		†					
	28x28	(12, 12)	73x73	(36, 36)			
3 x ResBlock(k=3, s=1)							
	28x28	(12, 12)	67x67	(33, 33)	21	18 †	2nd Layer of Interest FR – FO[21:-18, 21:-18]
ResBlock(k=3, s=2)		†					
	14x14	(6, 6)	33x33	(16, 16)			
5 x ResBlock(k=3, s=1)							
	14x14	(6, 6)	23x23	(11, 11)	5	4 †	3rd Layer of Interest FR – FO[5:-4. 5:-4]
ResBlock(k=3, s=2)		†					
	7x7	(3, 3)	11x11	(3, 3)			
2 x ResBlock(k=3, s=1)							
	7x7	(3, 3)	7x7	(3, 3)	0	0	4th Layer of Interest FR – FO
AdaptiveAvgPool2d							
	1x1						

Figure 6. **The architecture for VGG19 and ResNet50 used in the paper.** We mark the calculation of optimal padding in orange arrows and principal point in blue arrows. We label the layers of interest that are used in the paper. The red † indicates where a principal point shift is identified.

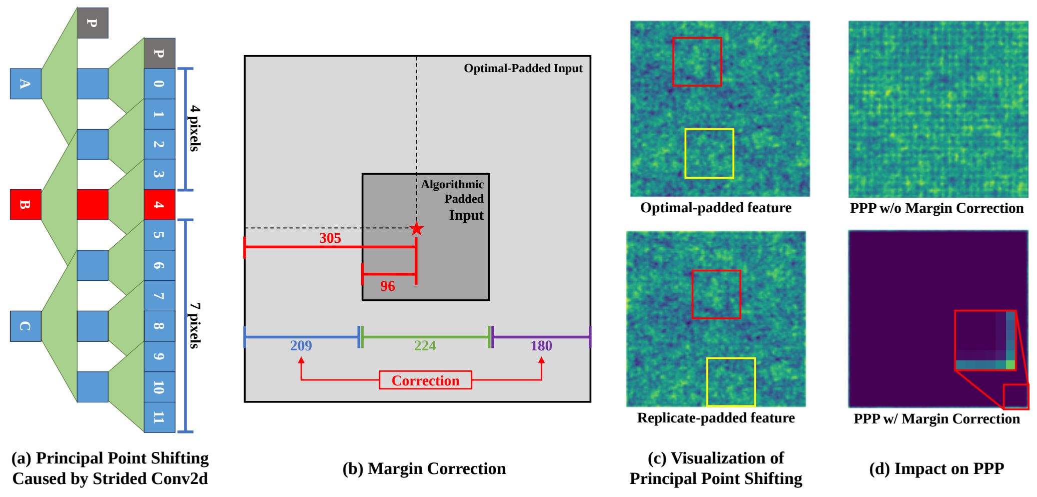


Figure 7. **Principal point shift.** (a) The stride-2 Conv2d only pads on one side, causing the principal point shift (red squares) in earlier layers. (b) Such a shift requires careful margin correction while aligning algorithmically-padded and optimally-padded features (we describe the details of point shift in Appendix A). (c) The shift is visible in the feature space (marked with red and yellow boxes). (d) It is crucial to correct the principal point shift while measuring PPP. The PPP calculation involves pixel-wise distance functions, which are not robust to spatial shifts (Zhang et al., 2018).

A.2. PPP Feature Misalignment

There are several pitfalls in visualizing and quantifying PPP. We identify two critical pitfalls from the architectures we implemented. However, these may not be sufficient to cover all potential issues while integrated into other architectures. Therefore one must be alerted to any unusual behavior (e.g., Figure 2(d) in the main paper) throughout their implementation.

Principal point shifting. Conv2d has a hidden behavior that few people are aware of, the operation is one-pixel skewed while applying a stride-two Conv2d on even-shaped features. To understand how the one-pixel shift happens, we first define the principal point of a feature map. We first define the principal point of the last feature map as the center pixel (note that we define it as the middle-point between the center-two pixels in case the last feature size is even). Then, we recursively define the principal point of the $(N-1)$-th layer as the pixel that positions at the center of the Conv2d receptive field that mainly forms the principal point of the N-th layer. In the case of optimally-padded features, the principal points in every layer are the center of the feature map. But, as shown in Figure 2(a), the principal point of algorithmically-padded features will have a one-pixel shift when a stride-2 convolution is applied to even-shaped features, which can be further amplified as more layers stack up. Such a skew causes the principal points of algorithmically-padded features shift several pixels away from the principal points of optimally-padded features. As PPP metrics use pixel-wise subtraction to distinguish the image content from PPP, the misalignment becomes a critical issue, since the image contents are no longer aligned and subtractable.

In Figure 6, we show the procedure of calculating the principal point in blue arrows and marking the values impacted by principal point shift with red †. For the ResNet50 architecture, the principal point shift accumulates to $16 (= 224/2 - 96)$ pixels in the early layers.

Fortunately, such a displacement can be fixed by adding corrections to how we calculate the feature margins. As shown in Figure 2(b), the concept of the margin correction is to make the two principal points overlapping each other after adding the margin. In the example, the left-right margins are corrected to $(209, 180)$ (instead of the more intuitive choice of $(195, 194)$ or $(194.5, 194.6)$).

We also show how the principal point shift visually looking like in Figure 2(c), notice the patterns have right-bottom shifted 16 pixels. As shown in Figure 2(d), failing to identify the principal point shift will result in checkerboard artifacts while calculating PPP, and adding correction eliminates the artifacts.

Maxpooling misalignment. This is a hypothetical condition that may potentially happen but has not been observed in the three architectures we tested. Consider a case of a Maxpooling layer of window size 2 and stride 2, the sliding windows of each pooling operation have no overlap, therefore the initial index of the first sliding window solely determines the spatial location of all sliding windows. Accordingly, there is a chance that the initial condition of the optimally-padded features causes all of its sliding windows to be one-pixel misaligned to the algorithmically-padded features. Fortunately, the condition can be easily determined by calculating the top and left margins of the feature alignment (similar to the aforementioned principal point shift calculation). For the case of a Maxpooling layer of window size 2 and stride 2, the misalignment will not happen if the top and left margins are even numbers, and that is exactly the case for VGG19 and ResNet50, as shown in Figure 6.

A.3. Randn Padding

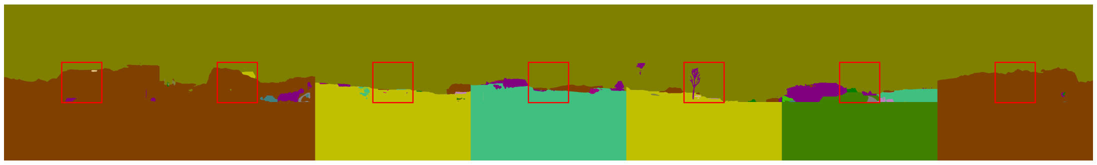
A critical implementation detail is that such a padding scheme must be applied before activation functions. Since the paddings are based on the distribution within sliding windows, activation functions such as ReLU, which clamps all negative values, can discard a significant amount of information beforehand. Instead of the traditional use of padding-convolution-normalization-activation, we modify the order to convolution-normalization-padding-activation. Note that such a change of order does not affect the behavior or results of other padding schemes.

B. The Full Experimental Results of Border Handling Variants (BHV) Test

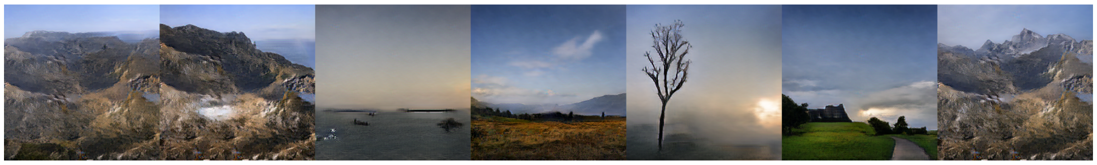
Table 5. **Background color as a critical confounding variable in BHV test.** We show that using a grey background similar to Figure 2 leads to discrepant results. The standard deviations are reported among 10 individual trials. We mark the best performance in green, and the worst two in red.

Padding	F-Conv?	Black Background				Grey Background			
		Similar (%)	Dissimilar (%)	Diff (%)	Inconsistency (%)	Similar (%)	Dissimilar (%)	Diff (%)	Inconsistency (%)
Zeros	N	$99.83_{\pm 0.00}$	$3.21_{\pm 8.35}$	-87.68	$95.81_{\pm 2.07}$	$100.00_{\pm 0.00}$	$4.96_{\pm 5.93}$	-95.04	$97.85_{\pm 4.55}$
	Y	$89.24_{\pm 0.98}$	$89.24_{\pm 0.98}$	0.00	$18.02_{\pm 8.08}$	$100.00_{\pm 0.00}$	$4.77_{\pm 6.52}$	-95.23	$96.79_{\pm 7.13}$
Circular	N	$80.31_{\pm 3.23}$	$80.31_{\pm 3.23}$	0.00	$34.25_{\pm 8.32}$	$72.75_{\pm 0.96}$	$72.75_{\pm 0.96}$	0.00	$26.30_{\pm 5.55}$
	Y	$99.20_{\pm 0.23}$	$93.14_{\pm 2.88}$	-6.06	$18.48_{\pm 3.55}$	$98.26_{\pm 0.50}$	$92.40_{\pm 4.23}$	-5.87	$28.67_{\pm 6.18}$
Reflect	N	$100.00_{\pm 0.00}$	$15.67_{\pm 12.72}$	-84.33	$91.18_{\pm 13.19}$	$100.00_{\pm 0.00}$	$19.96_{\pm 13.54}$	-80.04	$90.33_{\pm 11.95}$
	Y	$100.00_{\pm 0.00}$	$11.70_{\pm 15.38}$	-88.30	$97.33_{\pm 6.16}$	$100.00_{\pm 0.00}$	$17.16_{\pm 12.19}$	-82.84	$98.13_{\pm 3.44}$
Replicate	N	$100.00_{\pm 0.00}$	$43.39_{\pm 11.42}$	-56.61	$75.32_{\pm 8.20}$	$100.00_{\pm 0.00}$	$33.16_{\pm 6.42}$	-66.83	$84.09_{\pm 6.47}$
	Y	$98.32_{\pm 0.39}$	$93.65_{\pm 1.36}$	-4.67	$32.60_{\pm 4.97}$	$97.17_{\pm 0.48}$	$94.99_{\pm 1.20}$	-2.18	$32.15_{\pm 5.11}$
Randn	N	$100.00_{\pm 0.00}$	$10.31_{\pm 12.56}$	-89.70	$94.88_{\pm 5.55}$	$99.97_{\pm 0.13}$	$35.47_{\pm 10.82}$	-64.50	$83.59_{\pm 8.48}$
	Y	$100.00_{\pm 0.00}$	$20.80_{\pm 14.15}$	-79.20	$92.54_{\pm 8.37}$	$77.28_{\pm 16.13}$	$66.70_{\pm 11.58}$	-10.59	$45.70_{\pm 20.62}$
No-pad	-	$100.00_{\pm 0.00}$	$3.21_{\pm 8.35}$	-96.79	$95.81_{\pm 2.07}$	$100.00_{\pm 0.00}$	$30.07_{\pm 4.06}$	-69.93	$81.30_{\pm 2.44}$

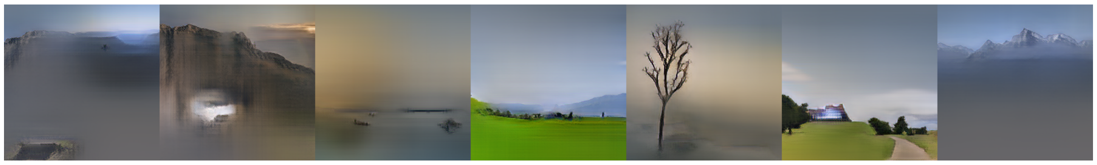
C. Examples of SPADE Degradation Due to The Change of Field-of-View



(a) Input semantic map condition (Full: 1024^2; red bounding box: 256^2)



(b) Synthetic results using only 256^2 bounding box region.



(c) Synthetic results using full 1024^2 input, then crop back to 256^2

Figure 8. **SPADE degradation due to change of field-of-view.** The SPADE model was trained at 256^2 pixels. The first row shows the input semantic segmentation map at 1024^2 pixels (four times field-of-view at the same resolution). The second row shows the synthesized results using 256^2 center crop of the inputs. The third row shows the synthesized results using 1024^2 inputs, then center-crop back to 256^2 after the model inference.

D. The Full Table of Comparing PPP with PosENet

Table 6. **Comparing PosENet and our PPP metric. Most of the PosENet results are indistinguishable due to high variation.** The standard deviation is computed over five different pretrained models. We report MAE metric for both PosENet and our PPP, use 2D Gaussian as PosENet reconstruction pattern, and measure PPP-MAE at the 4th layer of interest. (↑) indicates a higher value corresponds to stronger positional information (vice versa for (↓)). For each group of pretrained models, we label the strongest positional information response with red, and the experiments within its standard deviation range with blue. A good metric should have red entries concentrated under a single padding scheme and a few blue entries.

Model	Padding	Eval Dataset	PosENet		PPP-MAE(ours) (↑)	Performance (%)
			SPC (↑)	MAE (↓)		
VGG-19	Zeros	GMap	$0.107_{\pm 0.128}$	$0.196_{\pm 0.006}$	$0.0176_{\pm 0.0005}$	$74.0972_{\pm 0.0870}$
		InfinityGAN-flickr	$0.368_{\pm 0.116}$	$0.183_{\pm 0.007}$	$0.0163_{\pm 0.0006}$	
		InfinityGAN-tower	$0.492_{\pm 0.106}$	$0.173_{\pm 0.010}$	$0.0179_{\pm 0.0001}$	
		ImageNet-val	$0.237_{\pm 0.178}$	$0.190_{\pm 0.009}$	$0.0164_{\pm 0.0003}$	
		MS-COCO	$0.200_{\pm 0.173}$	$0.192_{\pm 0.009}$	$0.0173_{\pm 0.0002}$	
		PASCAL-S	$0.081_{\pm 0.145}$	$0.197_{\pm 0.006}$	$0.0163_{\pm 0.0002}$	
	Circular	GMap	$0.098_{\pm 0.139}$	$0.197_{\pm 0.007}$	$0.0158_{\pm 0.0006}$	$74.4716_{\pm 0.0863}$
		InfinityGAN-flickr	$0.323_{\pm 0.147}$	$0.185_{\pm 0.009}$	$0.0137_{\pm 0.0004}$	
		InfinityGAN-tower	$0.460_{\pm 0.102}$	$0.176_{\pm 0.009}$	$0.0184_{\pm 0.0005}$	
		ImageNet-val	$0.175_{\pm 0.174}$	$0.193_{\pm 0.008}$	$0.0161_{\pm 0.0003}$	
		MS-COCO	$0.148_{\pm 0.165}$	$0.194_{\pm 0.008}$	$0.0167_{\pm 0.0003}$	
		PASCAL-S	$0.083_{\pm 0.164}$	$0.197_{\pm 0.007}$	$0.0154_{\pm 0.0003}$	
	Reflect	GMap	$0.109_{\pm 0.139}$	$0.196_{\pm 0.007}$	$0.0158_{\pm 0.0002}$	$74.0516_{\pm 0.0621}$
		InfinityGAN-flickr	$0.343_{\pm 0.132}$	$0.185_{\pm 0.008}$	$0.0146_{\pm 0.0008}$	
		InfinityGAN-tower	$0.460_{\pm 0.113}$	$0.177_{\pm 0.009}$	$0.0168_{\pm 0.0005}$	
		ImageNet-val	$0.183_{\pm 0.193}$	$0.193_{\pm 0.009}$	$0.0157_{\pm 0.0004}$	
		MS-COCO	$0.173_{\pm 0.170}$	$0.193_{\pm 0.0007}$	$0.0165_{\pm 0.0002}$	
		PASCAL-S	$0.102_{\pm 0.182}$	$0.196_{\pm 0.0008}$	$0.0153_{\pm 0.0003}$	
	Replicate	GMap	$0.084_{\pm 0.137}$	$0.197_{\pm 0.006}$	$0.0144_{\pm 0.0009}$	$73.9964_{\pm 0.1079}$
		InfinityGAN-flickr	$0.356_{\pm 0.111}$	$0.184_{\pm 0.007}$	$0.0128_{\pm 0.0012}$	
		InfinityGAN-tower	$0.498_{\pm 0.111}$	$0.173_{\pm 0.010}$	$0.0156_{\pm 0.0006}$	
		ImageNet-val	$0.229_{\pm 0.181}$	$0.191_{\pm 0.009}$	$0.0143_{\pm 0.0006}$	
		MS-COCO	$0.209_{\pm 0.169}$	$0.192_{\pm 0.008}$	$0.0149_{\pm 0.0006}$	
		PASCAL-S	$0.110_{\pm 0.176}$	$0.196_{\pm 0.008}$	$0.0139_{\pm 0.0004}$	
	Randn	GMap	$0.125_{\pm 0.154}$	$0.195_{\pm 0.006}$	$0.0182_{\pm 0.0012}$	$73.7716_{\pm 0.0758}$
		InfinityGAN-flickr	$0.374_{\pm 0.137}$	$0.185_{\pm 0.007}$	$0.0167_{\pm 0.0008}$	
		InfinityGAN-tower	$0.421_{\pm 0.161}$	$0.181_{\pm 0.010}$	$0.0186_{\pm 0.0012}$	
		ImageNet-val	$0.204_{\pm 0.188}$	$0.192_{\pm 0.008}$	$0.0173_{\pm 0.0008}$	
		MS-COCO	$0.153_{\pm 0.180}$	$0.194_{\pm 0.007}$	$0.0182_{\pm 0.0008}$	
		PASCAL-S	$0.099_{\pm 0.201}$	$0.196_{\pm 0.008}$	$0.0166_{\pm 0.0008}$	
	NoPad	GMap	$0.001_{\pm 0.239}$	$0.204_{\pm 0.013}$	$0.0000_{\pm 0.0000}$	$62.0396_{\pm 0.0830}$
		InfinityGAN-flickr	$0.303_{\pm 0.192}$	$0.187_{\pm 0.012}$	$0.0000_{\pm 0.0000}$	
		InfinityGAN-tower	$0.516_{\pm 0.139}$	$0.172_{\pm 0.014}$	$0.0000_{\pm 0.0000}$	
		ImageNet-val	$0.048_{\pm 0.241}$	$0.200_{\pm 0.011}$	$0.0000_{\pm 0.0000}$	
		MS-COCO	$0.031_{\pm 0.231}$	$0.200_{\pm 0.010}$	$0.0000_{\pm 0.0000}$	
		PASCAL-S	$0.033_{\pm 0.257}$	$0.202_{\pm 0.013}$	$0.0000_{\pm 0.0000}$	

Table 7. **Comparing PosENet and our PPP metric. Most of the PosENet results are indistinguishable due to high variation.** The standard deviation is computed over five different pretrained models. We report MAE metric for both PosENet and our PPP, use 2D Gaussian as PosENet reconstruction pattern, and measure PPP-MAE at the 4th layer of interest. (↑) indicates a higher value corresponds to stronger positional information (vice versa for (↓)). For each group of pretrained models, we label the strongest positional information response with red, and the experiments within its standard deviation range with blue. A good metric should have red entries concentrated under a single padding scheme and a few blue entries.

Model	Padding	Eval Dataset	PosENet		PPP-MAE(ours) (↑)	Performance (%)
			SPC (↑)	MAE (↓)		
ResNet50	Zeros	GMap	$0.191_{\pm 0.188}$	$0.193_{\pm 0.008}$	$0.0162_{\pm 0.0012}$	$75.6856_{\pm 0.0924}$
		InfinityGAN-flickr	$0.682_{\pm 0.107}$	$0.152_{\pm 0.019}$	$0.0137_{\pm 0.0004}$	
		InfinityGAN-tower	$0.721_{\pm 0.077}$	$0.144_{\pm 0.017}$	$0.0153_{\pm 0.0013}$	
		ImageNet-val	$0.553_{\pm 0.194}$	$0.170_{\pm 0.016}$	$0.0143_{\pm 0.0005}$	
		MS-COCO	$0.465_{\pm 0.208}$	$0.179_{\pm 0.014}$	$0.0146_{\pm 0.0002}$	
		PASCAL-S	$0.259_{\pm 0.221}$	$0.190_{\pm 0.010}$	$0.0148_{\pm 0.0003}$	
	Circular	GMap	$0.398_{\pm 0.115}$	$0.197_{\pm 0.007}$	$0.0188_{\pm 0.0016}$	$76.1432_{\pm 0.1026}$
		InfinityGAN-flickr	$0.628_{\pm 0.084}$	$0.159_{\pm 0.013}$	$0.0178_{\pm 0.0005}$	
		InfinityGAN-tower	$0.585_{\pm 0.105}$	$0.165_{\pm 0.014}$	$0.0189_{\pm 0.0012}$	
		ImageNet-val	$0.397_{\pm 0.233}$	$0.182_{\pm 0.014}$	$0.0194_{\pm 0.0003}$	
		MS-COCO	$0.348_{\pm 0.238}$	$0.185_{\pm 0.014}$	$0.0203_{\pm 0.0002}$	
		PASCAL-S	$0.232_{\pm 0.244}$	$0.191_{\pm 0.011}$	$0.0199_{\pm 0.0006}$	
	Reflect	GMap	$0.197_{\pm 0.185}$	$0.192_{\pm 0.008}$	$0.0150_{\pm 0.0004}$	$75.5068_{\pm 0.1213}$
		InfinityGAN-flickr	$0.594_{\pm 0.096}$	$0.169_{\pm 0.012}$	$0.0134_{\pm 0.0009}$	
		InfinityGAN-tower	$0.667_{\pm 0.087}$	$0.153_{\pm 0.016}$	$0.0157_{\pm 0.0002}$	
		ImageNet-val	$0.493_{\pm 0.206}$	$0.178_{\pm 0.013}$	$0.0137_{\pm 0.0005}$	
		MS-COCO	$0.401_{\pm 0.230}$	$0.182_{\pm 0.015}$	$0.0138_{\pm 0.0005}$	
		PASCAL-S	$0.250_{\pm 0.223}$	$0.190_{\pm 0.010}$	$0.0139_{\pm 0.0004}$	
	Replicate	GMap	$0.249_{\pm 0.192}$	$0.189_{\pm 0.009}$	$0.0138_{\pm 0.0003}$	$75.6122_{\pm 0.0911}$
		InfinityGAN-flickr	$0.700_{\pm 0.095}$	$0.147_{\pm 0.018}$	$0.0114_{\pm 0.0003}$	
		InfinityGAN-tower	$0.726_{\pm 0.069}$	$0.142_{\pm 0.016}$	$0.0142_{\pm 0.0007}$	
		ImageNet-val	$0.536_{\pm 0.194}$	$0.172_{\pm 0.015}$	$0.0127_{\pm 0.0008}$	
		MS-COCO	$0.458_{\pm 0.209}$	$0.179_{\pm 0.014}$	$0.0129_{\pm 0.0003}$	
		PASCAL-S	$0.320_{\pm 0.237}$	$0.186_{\pm 0.012}$	$0.0128_{\pm 0.0003}$	
	Randn	GMap	$0.210_{\pm 0.192}$	$0.191_{\pm 0.009}$	$0.0147_{\pm 0.0007}$	$75.3076_{\pm 0.1016}$
		InfinityGAN-flickr	$0.566_{\pm 0.100}$	$0.171_{\pm 0.011}$	$0.0122_{\pm 0.0011}$	
		InfinityGAN-tower	$0.714_{\pm 0.068}$	$0.142_{\pm 0.015}$	$0.0153_{\pm 0.0004}$	
		ImageNet-val	$0.416_{\pm 0.207}$	$0.182_{\pm 0.013}$	$0.0141_{\pm 0.0004}$	
		MS-COCO	$0.430_{\pm 0.242}$	$0.178_{\pm 0.017}$	$0.0142_{\pm 0.0005}$	
		PASCAL-S	$0.336_{\pm 0.231}$	$0.186_{\pm 0.013}$	$0.0139_{\pm 0.0004}$	

Table 8. **Comparing PosENet and our PPP metric. Most of the PosENet results are indistinguishable due to high variation.** The standard deviation is computed over five different pretrained models. We report MAE metric for both PosENet and our PPP, use 2D Gaussian as PosENet reconstruction pattern, and measure PPP-MAE at the 4th layer of interest. (↑) indicates a higher value corresponds to stronger positional information (vice versa for (↓)). For each group of pretrained models, we label the strongest positional information response with red, and the experiments within its standard deviation range with blue. A good metric should have red entries concentrated under a single padding scheme and a few blue entries.

Model	Padding	Eval Dataset	PosENet		PPP-MAE(ours) (↑)	Performance (%)
			SPC (↑)	MAE (↓)		
SOD (PiCANet)	Zeros	GMap	$0.156_{\pm 0.212}$	$0.201_{\pm 0.017}$	$0.0049_{\pm 0.0001}$	$62.69_{\pm 0.0015}$
		InfinityGAN-flickr	$0.365_{\pm 0.140}$	$0.184_{\pm 0.012}$	$0.0036_{\pm 0.0001}$	
		InfinityGAN-tower	$0.449_{\pm 0.120}$	$0.179_{\pm 0.013}$	$0.0032_{\pm 0.0001}$	
		ImageNet-val	$0.288_{\pm 0.259}$	$0.189_{\pm 0.018}$	$0.0034_{\pm 0.0001}$	
		MS-COCO	$0.265_{\pm 0.237}$	$0.190_{\pm 0.016}$	$0.0033_{\pm 0.0000}$	
		PASCAL-S	$0.307_{\pm 0.240}$	$0.188_{\pm 0.016}$	$0.0031_{\pm 0.0000}$	
	Circular	GMap	$0.011_{\pm 0.209}$	$0.207_{\pm 0.014}$	$0.0062_{\pm 0.0001}$	$62.60_{\pm 0.0009}$
		InfinityGAN-flickr	$0.329_{\pm 0.133}$	$0.187_{\pm 0.012}$	$0.0068_{\pm 0.0001}$	
		InfinityGAN-tower	$0.398_{\pm 0.115}$	$0.182_{\pm 0.011}$	$0.0050_{\pm 0.0002}$	
		ImageNet-val	$0.249_{\pm 0.274}$	$0.191_{\pm 0.019}$	$0.0050_{\pm 0.0001}$	
		MS-COCO	$0.208_{\pm 0.244}$	$0.194_{\pm 0.016}$	$0.0049_{\pm 0.0000}$	
		PASCAL-S	$0.249_{\pm 0.248}$	$0.192_{\pm 0.017}$	$0.0048_{\pm 0.0000}$	
	Reflect	GMap	$0.062_{\pm 0.210}$	$0.205_{\pm 0.016}$	$0.0053_{\pm 0.0001}$	$62.43_{\pm 0.0022}$
		InfinityGAN-flickr	$0.322_{\pm 0.133}$	$0.188_{\pm 0.013}$	$0.0030_{\pm 0.0001}$	
		InfinityGAN-tower	$0.396_{\pm 0.125}$	$0.183_{\pm 0.013}$	$0.0039_{\pm 0.0001}$	
		ImageNet-val	$0.290_{\pm 0.267}$	$0.190_{\pm 0.020}$	$0.0040_{\pm 0.0001}$	
		MS-COCO	$0.239_{\pm 0.250}$	$0.193_{\pm 0.017}$	$0.0040_{\pm 0.0000}$	
		PASCAL-S	$0.305_{\pm 0.242}$	$0.190_{\pm 0.019}$	$0.0035_{\pm 0.0000}$	
	Replicate	GMap	$0.071_{\pm 0.215}$	$0.204_{\pm 0.016}$	$0.0043_{\pm 0.0002}$	$62.55_{\pm 0.0013}$
		InfinityGAN-flickr	$0.335_{\pm 0.139}$	$0.186_{\pm 0.012}$	$0.0023_{\pm 0.0001}$	
		InfinityGAN-tower	$0.409_{\pm 0.120}$	$0.182_{\pm 0.012}$	$0.0032_{\pm 0.0001}$	
		ImageNet-val	$0.312_{\pm 0.264}$	$0.188_{\pm 0.019}$	$0.0032_{\pm 0.0000}$	
		MS-COCO	$0.260_{\pm 0.246}$	$0.191_{\pm 0.016}$	$0.0030_{\pm 0.0001}$	
		PASCAL-S	$0.340_{\pm 0.237}$	$0.187_{\pm 0.019}$	$0.0026_{\pm 0.0000}$	
	Randn	GMap	$0.002_{\pm 0.244}$	$0.202_{\pm 0.009}$	$0.0001_{\pm 0.0000}$	$25.70_{\pm 0.0022}$
		InfinityGAN-flickr	$0.228_{\pm 0.173}$	$0.197_{\pm 0.010}$	$0.0001_{\pm 0.0000}$	
		InfinityGAN-tower	$0.212_{\pm 0.148}$	$0.200_{\pm 0.011}$	$0.0001_{\pm 0.0000}$	
		ImageNet-val	$0.108_{\pm 0.397}$	$0.203_{\pm 0.021}$	$0.001_{\pm 0.0000}$	
		MS-COCO	$0.176_{\pm 0.321}$	$0.200_{\pm 0.018}$	$0.001_{\pm 0.0000}$	
		PASCAL-S	$0.155_{\pm 0.329}$	$0.203_{\pm 0.020}$	$0.001_{\pm 0.0000}$	
	NoPad	GMap	$0.000_{\pm 0.264}$	$0.211_{\pm 0.019}$	$0.0000_{\pm 0.0000}$	$47.59_{\pm 0.0013}$
		InfinityGAN-flickr	$0.454_{\pm 0.194}$	$0.178_{\pm 0.021}$	$0.0000_{\pm 0.0000}$	
		InfinityGAN-tower	$0.520_{\pm 0.167}$	$0.172_{\pm 0.020}$	$0.0000_{\pm 0.0000}$	
		ImageNet-val	$0.072_{\pm 0.282}$	$0.203_{\pm 0.018}$	$0.0000_{\pm 0.0000}$	
		MS-COCO	$0.065_{\pm 0.277}$	$0.203_{\pm 0.017}$	$0.0000_{\pm 0.0000}$	
		PASCAL-S	$0.086_{\pm 0.283}$	$0.206_{\pm 0.020}$	$0.0000_{\pm 0.0000}$	

E. More PPP Visualizations

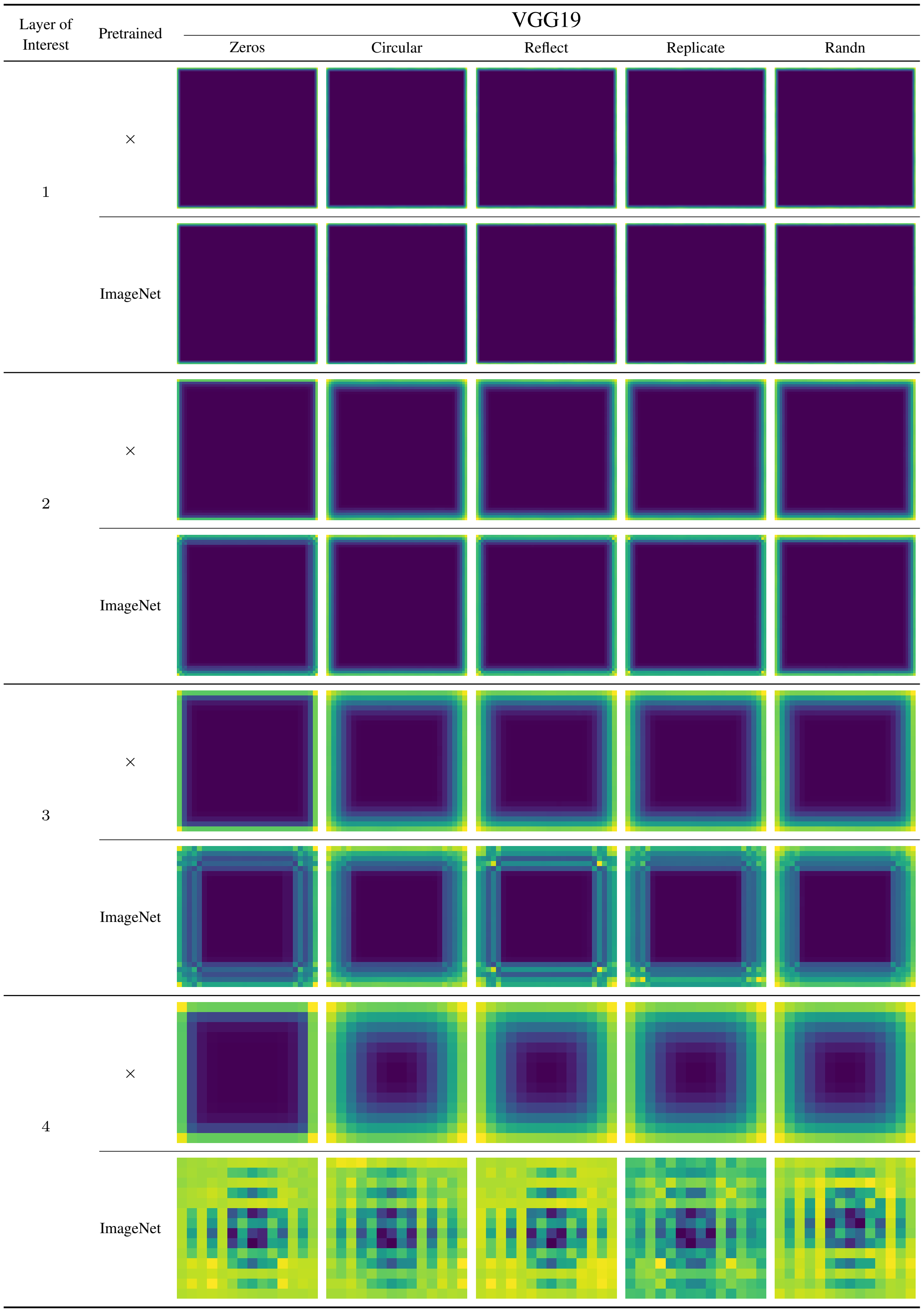


Figure 9. **Visualization of Position-Information Pattern from Padding (PPP).** The visualizations are calculated based on Eq. 3 over 480 GMap samples. The results show that the pretrained model significantly reinforces PPP compared to randomly initialized networks. Note that each image is normalized to $[0, 1]$ separately, therefore the colors between images are not comparable.

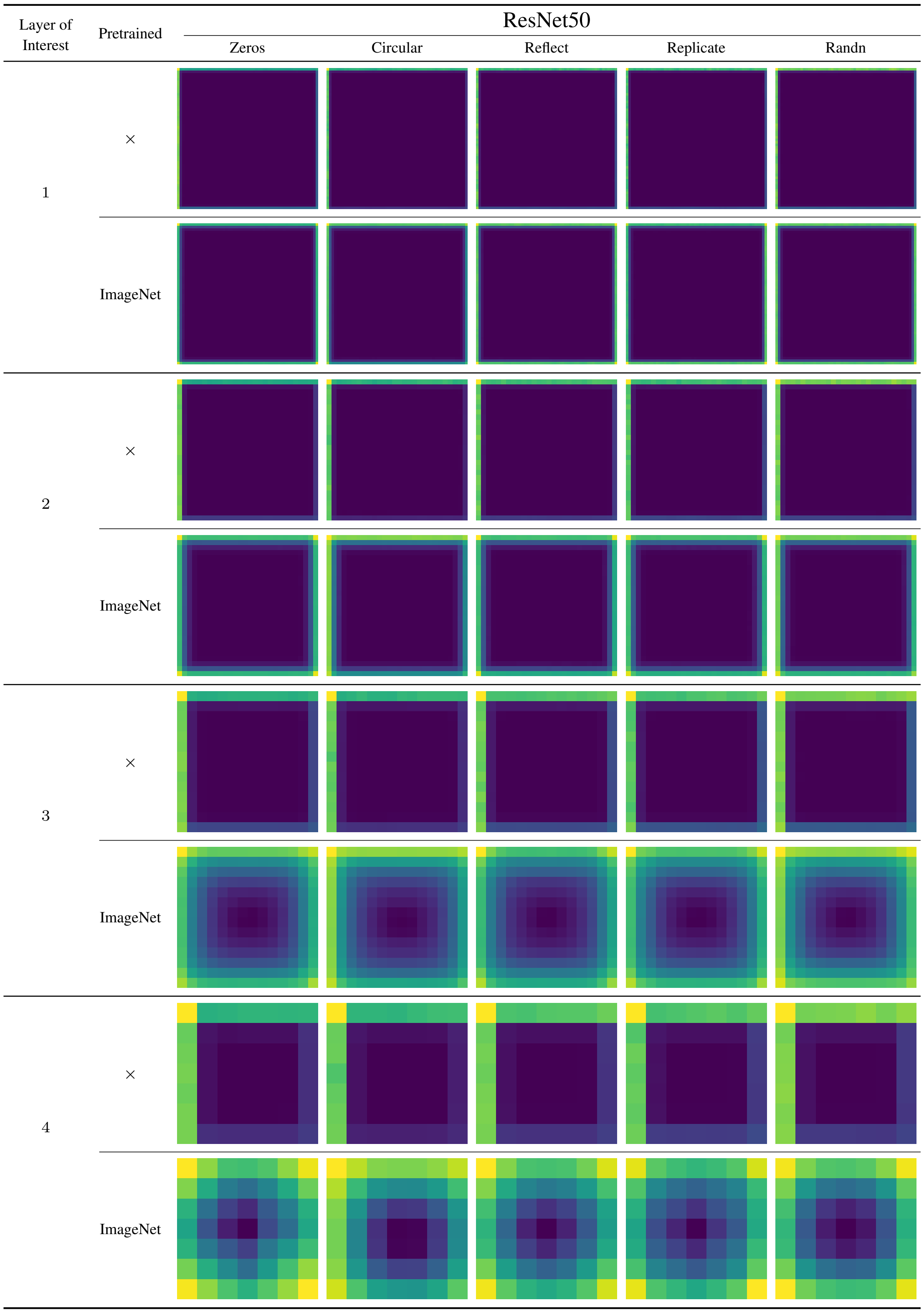


Figure 10. **Visualization of Position-Information Pattern from Padding (PPP).** The visualizations are calculated based on Eq. 3 over 480 GMap samples. The results show that the pretrained model significantly reinforces PPP compared to randomly initialized networks. Note that each image is normalized to $[0, 1]$ separately, therefore the colors between images are not comparable.

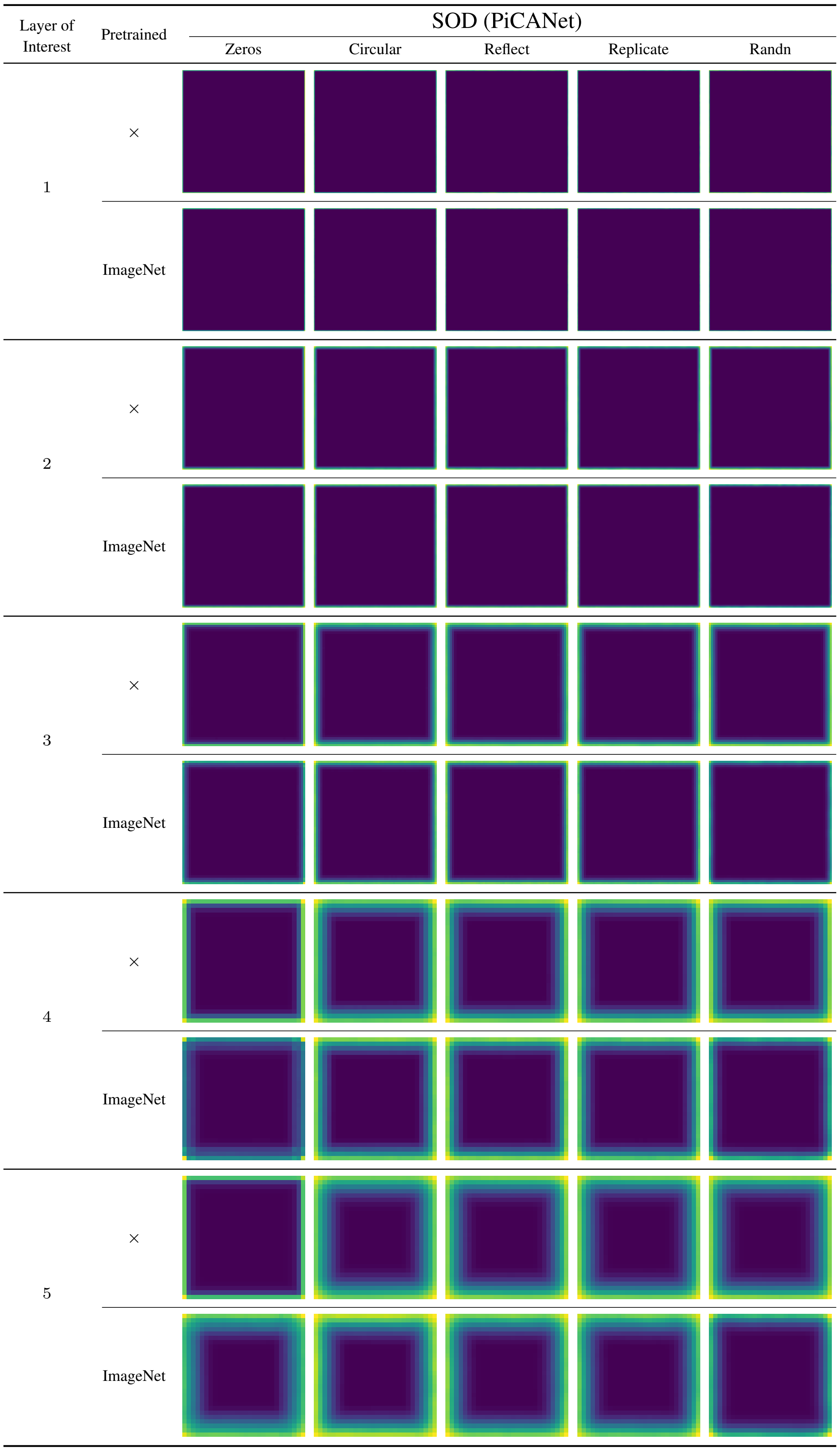


Figure 11. **Visualization of Position-Information Pattern from Padding (PPP).** The visualizations are calculated based on Eq. 3 over 480 GMap samples. The results show that the pretrained model significantly reinforces PPP compared to randomly initialized networks. Note that each image is normalized to $[0, 1]$ separately, therefore the colors between images are not comparable.