One-dimensional Path Convolution

Xuanshu Luo [1] **Martin Werner** [1]

Abstract

Two-dimensional (2D) convolutional kernels have dominated convolutional neural networks (CNNs) in image processing. While linearly scaling 1D convolution provides parameter efficiency, its naive integration into CNNs disrupts image locality, thereby degrading performance. This paper presents path convolution (PathConv), a novel CNN design exclusively with 1D operations, achieving ResNet-level accuracy using only 1/3 parameters. To obtain locality-preserving image traversal paths, we analyze Hilbert/Z-order paths and expose a fundamental trade-off: improved proximity for most pixels comes at the cost of excessive distances for other sacrificed ones to their neighbors. We resolve this issue by proposing path shifting, a succinct method to reposition sacrificed pixels. Using the randomized rounding algorithm, we show that three shifted paths are sufficient to offer better locality preservation than trivial raster scanning. To mitigate potential convergence issues caused by multiple paths, we design a lightweight path-aware channel attention mechanism to capture local intra-path and global inter-path dependencies. Experimental results further validate the efficacy of our method, establishing the proposed 1D PathConv as a viable backbone for efficient vision models.

1. Introduction

Convolutional Neural Networks (CNNs) have emerged as the predominant paradigm for computer vision tasks across diverse domains (LeCun et al., 1989; Fukushima, 1988; Krizhevsky et al., 2017; Girshick, 2015; Chen et al., 2018; Ravanbakhsh et al., 2016), with two-dimensional (2D) convolution serving as the fundamental operation. In comparison, 1D convolution operating sequentially along a single dimension is inadequate for image processing due to its inability to preserve the spatial continuity of adjacent pixels in both directions. Nonetheless, vision models still incorporate 1D convolution through factorized convolution (Rigamonti et al., 2013; Szegedy et al., 2016; 2017; Peng et al., 2017; Guo et al., 2022) or oriented kernels (Freeman et al., 1991; Weiler et al., 2018; Li et al., 2021; Kirchmeyer & Deng, 2023) to reduce computational complexity. The primary rationale arises from the architectural simplicity of 1D convolution scaling linearly with the kernel size rather than quadratically like 2D convolution.

These insights drive this paper's pursuit of an ostensibly paradoxical objective: constructing a vision model that exclusively utilizes 1D convolution to achieve superior parameter efficiency while simultaneously overcoming its intrinsic limitations to match the capabilities of conventional 2D CNNs. The sequential architecture comprising solely 1D operators facilitates pipelining implementation on hardware (Mukherjee & Mukhopadhyay, 2018; Cheng & Parhi, 2020; Narayanan et al., 2021), thereby substantially reducing end-to-end latency. Consequently, using only 1D convolution offers apparent advantages in computational efficiency. The primary challenge lies in enabling 1D CNNs to effectively interpret visual information.

This work employs only 1D operators to construct CNNs, necessitating 2D images to be flattened. Trivial image traversal by raster scanning disrupts the spatial locality of images, which is essential for detecting primitive visual features, such as edges, textures, and shapes. Unlike raster scanning, the Hilbert (Hilbert, 1935) and Z-order (Morton, 1966) curves are two alternative traversal *paths* with recognized spatial proximity preservation capability (Jagadish, 1990; Dai & Su, 2003; Wang et al., 2022; Wu et al., 2024), indicating their great potential to be the key ingredient to 1D CNNs to achieve comparable performance to 2D CNNs. However, this paper argues that the locality preservation properties of these two paths have a critical trade-off: compared to raster scanning, while the majority of pixels maintain closer proximity to their adjacent pixels in the resulting 1D pixel streams, the remaining pixels are sacrificed due to excessively long distances to neighbors, even sometimes leading to greater total distances than raster scanning. This finding persists for multiple image resolutions, demonstrating

[1]Professorship of Big Geospatial Data Management, Technical University of Munich, Germany. Correspondence to: Xuanshu Luo <xuanshu.luo@tum.de>, Martin Werner <martin.werner@tum.de>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

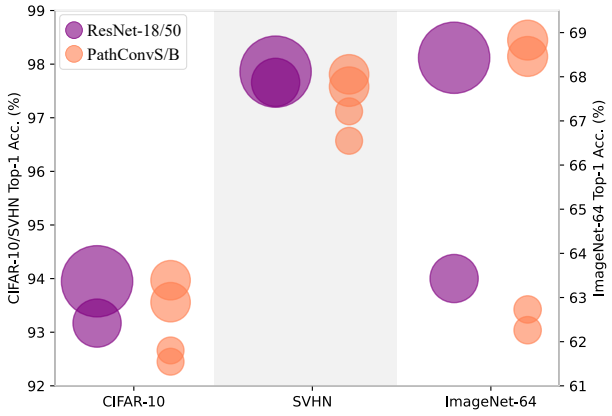


Figure 1. The performance comparison between PathConv and ResNet models on three datasets, with circle radii denoting the number of model parameters. PathConv models achieve ResNet-level performance using only 1/3 of the parameters.

that direct usage of both paths cannot globally guarantee better locality than raster scanning, inducing suboptimal performance of 1D CNNs relative to 2D architectures.

This paper first presents that the positions of these sacrificing pixels can be shifted by sampling from their spatially expanded variants, inspiring us to integrate a set of carefully selected shifted paths to ensure pixels at all positions are closer to their neighbors than raster scanning for at least one path. Meanwhile, limiting the number of paths is essential to avoid significant computational overhead, as input images must be traversed once for each path. Determining the minimal set of paths to ensure better locality preservation than raster scanning for all pixels is equivalent to the NP set cover problem (Vazirani, 2003). Using the randomized rounding algorithm (Bertsimas & Vohra, 1998), we show that merely three paths suffice for Hilbert and Z-order paths. A customized CUDA kernel further accelerates the image traversal procedure up to 73x, making its costs negligible.

Building upon this efficient image traversal layer, we propose *path convolution* models adapted from the macro design of depthwise separable convolution (Chollet, 2017) and inverted bottleneck (Sandler et al., 2018) but using exclusively 1D operators. Notably, multiple traversal paths require 1D kernels to process distinct regions of images concurrently, which presents challenges for model convergence. Hence, we propose a path-aware channel attention mechanism to capture both local intra-path and global inter-path dependencies to achieve dynamic focus on images, thereby reasonably enabling rapid receptive field growth. Experimental results demonstrate that our models achieve comparable performance to ResNet (He et al., 2016) while utilizing only 1/3 parameters, indicating that the proposed path convolution effectively maximizes the parameter efficiency of 1D kernels with minimal performance degradation.

2. Related Work

2.1. 1D convolution in CNNs

The linear scaling cost of 1D convolution with the kernel size presents compelling opportunities for efficient 2D CNN architectures. We separately discuss them based on their distinct theoretical foundations.

Factorized convolution. The core idea of factorized convolution builds on a fundamental principle from linear algebra that a 2D convolution kernel can be approximated or exactly decomposed into sequential 1D convolutions. For instance, a rank-1 separable kernel is equivalent to the product of consecutive horizontal and vertical 1D kernels (Rigamonti et al., 2013; Jaderberg et al., 2014; Jin et al., 2014), enabling a $k \times k$ kernel to be substituted by two 1D kernels with $2k$ parameters in total. Such parameter reduction becomes increasingly significant as k increases while maintaining identical receptive field sizes, so CNN architectures involving large kernel sizes widely adopt this idea (Szegedy et al., 2016; 2017; Peng et al., 2017; Guo et al., 2022; Huang et al., 2023). However, 2D kernels in CNNs do not inherently possess low-rank properties. Consequently, many methods have emerged to better approximate 2D kernels by learning low-rank kernels and applying singular value decomposition, aiming to minimize potential performance degradation (Denton et al., 2014; Lin et al., 2018; Yang et al., 2020).

Oriented convolution. Unlike factorized convolution, oriented kernels place parameters along multiple angles beyond solely horizontal and vertical orientations (Li et al., 2021; Kirchmeyer & Deng, 2023). The underlying motivation could be illustrated by a trivial example: a full-rank 2D kernel denoted by a diagonal matrix is hard to decompose into horizontal and vertical 1D kernels but can be directly represented by a single 45°oriented 1D kernel. However, these methods generally require dedicated software and hardware implementation for efficiency and optimization, as popular deep-learning libraries do not natively support them. A conceptually related technique is steerable filters, which enhance their expressiveness by learning rotation-equivalence/invariance features, which are proven effective in capturing directional characteristics like textures and edges (Freeman & Adelson, 1991; Cohen & Welling, 2016; Worrall et al., 2017; Weiler et al., 2018). These methods are usually more complex than oriented kernels and do not necessarily use 1D convolution.

In comparison, the proposed path convolution model comprises only 1D operations to facilitate pipelining and maximize the parameter efficiency of 1D kernels, processing pixel streams flattened from input images, which differentiates this paper from all methods above incorporating 1D convolution while maintaining 2D input/output signatures. However, images inherently have a spatial structure where

proximate pixels have strong correlations and form meaningful patterns. Flattening images into 1D pixel streams disrupts these crucial locality features. To address this issue, this paper uses traversal paths based on Hilbert and Z-order curves for better locality preservation than raster scans.

2.2. Hilbert and Z-order curves

Hilbert and Z-order curves exemplify space-filling curves (Peano, 1890) passing through all unit squares (pixels) exactly once in two- or higher-dimensional discrete spaces with topological self-organizing properties, i.e., invariant geometric structure across multiple scales. Concretely, one can observe a pattern topologically equivalent to the complete curve at any segment for arbitrary magnification levels, differing only in scale and/or rotational orientation, as shown in Figure 2. This characteristic originates from their recursive definitions, creating a scale-invariant 1D-to-2D mapping inherently constraining the spatial distances of points in the output space relative to their input space proximity (Lempel & Ziv, 1986; Dai & Su, 2003). Hence, the self-organizing nature of Hilbert and Z-order curves enables their locality preservation properties for multiple resolutions, empowering applications in image compression (Ansari & Fineberg, 1992; Wang et al., 2022), databases (Kamel & Faloutsos, 1994; Castro et al., 2005), parallel computing (Yoo et al., 2003; Böhm et al., 2018), and point cloud processing (Chen et al., 2022; 2023; Wu et al., 2024).

However, this paper traverses images along paths defined by Hilbert and Z-order curves, mapping from 2D to 1D rather than the reverse, thus hardly ensuring equivalent strength in locality preservation. The fundamental limitation arises because while two points may be proximate in 2D space, 1D paths might traverse a long distance to connect them, particularly for pixels at (sub)quadrant boundaries, resulting in some spatially adjacent points being mapped farther apart in the 1D sequence than raster scanning, as depicted in Figure 3 (see Appendix A for more examples in multiple resolutions). Consequently, the direct application of Hilbert or Z-order paths for image traversal produces suboptimal results in image recognition tasks, as evidenced by our experiments (Table 4). In this paper, we mitigate this limitation by integrating multiple shifted paths sampled from their spatially expanded variants as elaborated in Section 3.2, thereby ensuring better locality preservation for every pixel than raster scanning in at least one path.

3. Locality-preserving Paths

This section aims to find a set of paths satisfying the *locality constraint*, wherein each pixel maintains shorter cumulative distances to its neighbors than its corresponding pixel in the raster scan path. Our locality measurement reveals that conventional Hilbert and Z-order paths possess imperfect

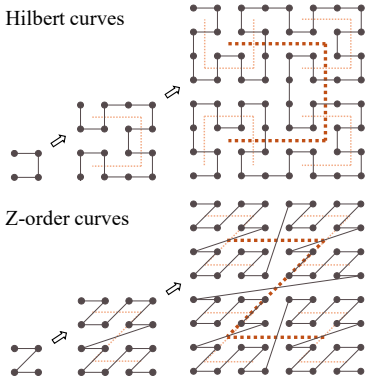


Figure 2. Hilbert and Z-order curves for multiple sizes. Hilbert curves repeat a U-shape pattern with rotations, whereas Z-order curves replicate a Z-shape pattern without rotations.

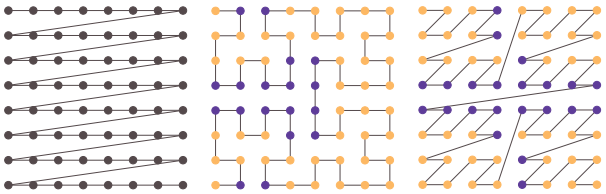


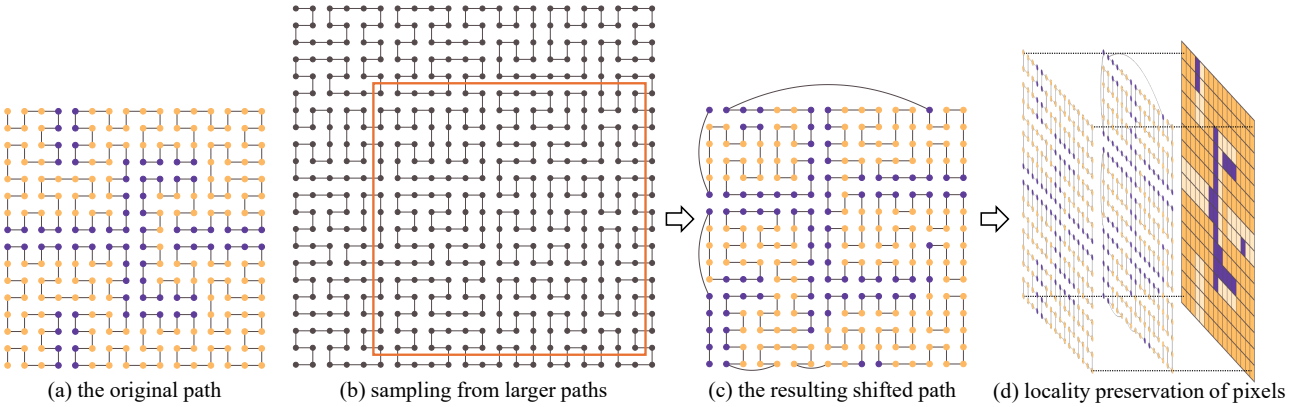
Figure 3. Pixel-wise comparison between (a) raster scan and (b) Hilbert / (c) Z-order paths. Pixels with shorter cumulative distances to all their neighbors than (a) are in **apricot**, otherwise **purple**.

locality preservation. To address this deficiency, we propose a concise path-shifting method to relocate pixels with long distances to their neighbors. While combining multiple shifted paths suffices the locality constraint, minimizing the number of paths remains essential for optimizing traversal efficiency. Our analysis indicates that determining the minimal set of paths fulfilling the locality constraint constitutes an NP problem. We use the randomized rounding algorithm to solve the problem and find that three paths suffice the locality constraint for commonly used image resolutions.

3.1. Locality measurements

A straightforward and convincing approach to assessing the locality preservation capability of a traversal path is to calculate the cumulative distances between each pixel and all its adjacent neighbors in resulting 1D pixel streams. Comparing to the raster scan path served as the baseline, except for comparing the total distances of all pixels in a path, pixel-wise comparisons are also taken into account by calculating the proportion of pixels with shorter distances to their neighbors at the same positions, denoted by P_{sd}.

Table 1 reports the locality measurements of Hilbert and Z-order paths versus raster scan paths for multiple resolutions. Although Hilbert paths guarantee that P_{sd} increases as the resolution grows, the total distances are contradic-



(a) the original path (b) sampling from larger paths (c) the resulting shifted path (d) locality preservation of pixels

Figure 4. The path shifting procedure. Given (a) $\mathcal{H}_{16}$ and $c = \{6, [1, 5], 0\}$, an expanded (b) $\mathcal{H}_{22}$ $(16 + 6)$ is firstly generated. Then, a 16×16 sampling window, with its bottom left coordinate at $[1, 5]$, is applied to (b) to obtain (c) $\mathcal{H}^c_{16}$. The coloring schema of Figure 3 is applied to (a) and (c). Jointly considering (a) and (c), (d) shows the locality preservation status of all pixels compared with $\mathcal{R}_{16}$, where (a) initially enables apricot pixels to have better locality preservation. (c) further enables light apricot pixels, demonstrating the effectiveness of the proposed path-shifting method for relocating sacrificed pixels. Purple pixels still have longer distances to their neighbors.

Table 1. Locality measurements of the raster scan, Hilbert, and Z-order paths for multiple resolutions.

Path	Resolution	Total Distance	P_{sd}
raster scan	32×32	1.88×10^5	-
Hilbert		2.15×10^5	79.10%
Z-order		1.80×10^5	74.61%
raster scan	64×64	1.54×10^6	-
Hilbert		1.80×10^6	80.96%
Z-order		1.49×10^6	81.35%
raster scan	128×128	1.24×10^7	-
Hilbert		1.48×10^7	86.51%
Z-order		1.22×10^7	84.50%
raster scan	256×256	1.00×10^8	-
Hilbert		1.20×10^8	89.04%
Z-order		0.99×10^8	89.17%

torily longer. This is because most pixels at the border of quadrant boundaries are *sacrificed* to keep the recursive, self-similar structures of Hilbert paths, necessitating non-local multi-step connections between spatially adjacent pixels located in different quadrants of recursive subdivisions, as previously shown in Figure 3. While Z-order paths maintain shorter total distances than the raster scan path, they fail to consistently provide higher P_{sd} for the same reason as Hilbert paths, not to mention the locality constraint implies $P_{sd} = 100\%$. Consequently, directly applying Hilbert and Z-order paths yields suboptimal spatial locality preservation, thereby constraining the model performance.

3.2. Path shifting

This paper introduces a simple method for repositioning sacrificed pixels in Hilbert and Z-order paths, eliminating their concentration at (sub)quadrant boundaries. Figure 4 illustrates the detailed procedure using a Hilbert path as an example. Given images of size $s \times s$, shifted paths are sampled from their spatially extended variants, with sizes equal to the sum of s and the padding size p. Subsequently, the sampling window of dimension $s \times s$ is positioned according to its bottom left coordinates $[i, j]$, where i, j are integers in range $[0, p]$. Besides, shifted paths could be rotated r degrees, where $r \in \{0, 90, 180, 270\}$, to achieve more diverse repositioning of sacrificed pixels. A configuration $c = \{p, [i, j], r\}$ thus specifies a path-shifting operation. Accordingly, we denote shifted Hilbert and Z-order paths of size $s \times s$ with configuration c as $\mathcal{H}^c_s$ and $\mathcal{Z}^c_s$, respectively. The original Hilbert and Z-order paths conform to this notation with c omitted, as all parameters equal 0, yielding $\mathcal{H}_s$ and $\mathcal{Z}_s$, respectively. Similarly, we have $\mathcal{R}_s$ to denote the raster scan path. The proposed shifting approach effectively relocates sacrificed pixels previously located at (sub)quadrant boundaries, as shown in Figure 4, revealing the feasibility of meeting the locality constraint. When a set of c generates shifted paths that fulfill the locality constraint, we denote it as $\mathbb{C}$. In other words, a set of paths $\mathcal{H}^{\mathbb{C}}_s$ or $\mathcal{Z}^{\mathbb{C}}_s$ can guarantee that there are no sacrificed pixels.

We only consider square images here for simplicity. Appendix B further indicates that this shifting method also works for non-square resolutions and substantially broadens the applicability of $\mathcal{Z}_s$ by removing its limitation of $s \in \{2^n \mid n \in \mathbb{N}\}$, where $\mathbb{N}$ is the set of natural numbers.

3.3. The minimal $\mathbb{C}$ satisfying the locality constraint

The cardinality of $\mathbb{C}$, denoted as $|\mathbb{C}|$, represents the number of shifted paths. While a large $|\mathbb{C}|$ trivially meets the locality constraint, it simultaneously introduces substantial overhead by traversing images for $|\mathbb{C}|$ times. Therefore, the ideal solution is determining $\mathbb{C}$ with the smallest possible

Table 2. The configurations denoted as $\mathbb{C}^*$ satisfying the locality constraint when $|\mathbb{C}| = 3$ for Hilbert and Z-order paths.

Path	s	Configurations ($\mathbb{C}^*$)
Hilbert	32	{0, [0, 0], 0} {24, [14, 1], 180} {18, [7, 0], 270}
	64	{0, [0, 0], 0} {7, [7, 0], 180} {40, [13, 1], 0}
Z-order	32	{32, [16, 26], 180} {32, [14, 16], 0} {32, [12, 20], 0}
	64	{64, [36, 32], 270} {64, [36, 34], 90} {64, [28, 32], 0}

cardinality satisfying the locality constraint. In Appendix C, we demonstrate that this problem is *polynomial-time reducible* to the NP *set cover problem* (Vazirani, 2003). Hence, existing algorithms and theoretical bounds of the set cover problem could be applied to our problem to find and evaluate satisfactory $\mathbb{C}$. We employ the randomized rounding algorithm (Raghavan & Tompson, 1987) trying to find the minimal $\mathbb{C}$ satisfying the locality constraint. First, the universe $\mathcal{C}$ must be specified given image size s^2. $|\mathcal{C}|$ should be sufficiently large to ensure adequate diversity in the distribution of sacrificed pixels while remaining constrained to maintain computational feasibility given the NP complexity. In each configuration $c = \{p, [i, j], r\}$, where $r \in \{0, 90, 180, 270\}$ and $[i, j]$ is determined by p with $i, j \in [0, p]$, only p requires detailed consideration.

Both $\mathcal{H}_s$ and $\mathcal{Z}_s$ are defined recursively and self-replicated when s is doubled. When $p \geq s$, the same $[i, j]$ derives similar shifted paths where the positions of sacrificed pixels exhibit no significant differences between $\mathcal{H}^c_{s+p}$ / $\mathcal{Z}^c_{s+p}$ and $\mathcal{H}^c_s$ / $\mathcal{Z}^c_s$ (see Appendix D). Consequently, $[1, s)$ is a reasonable parameter space for p. In practice, while $\mathcal{H}_s$ can directly apply this setting, $\mathcal{Z}_s$ only accept $s \in T$, where $T = \{2^n \mid n \in \mathbb{N}\}$. Hence, we set $p = t - s$ for $\mathcal{Z}_s$, where t is the smallest element in T larger than s. In this way, $\mathcal{C}$ could be determined given s for both kinds of paths. Solving by the randomized rounding algorithm, we find that $|\mathbb{C}| = 3$ sufficiently satisfies the locality constraint. The corresponding c are given in Table 2. Considering the theoretical approximation ratio of $\ln s^2 \approx 6.93$ and 8.32 for $s = 32$ and 64, respectively (Bertsimas & Vohra, 1998; Feige, 1998), $|\mathbb{C}| = 3$ is highly likely optimal, as one path cannot meet the locality constraint. We designate these $\mathbb{C}$ as $\mathbb{C}^*$ to represent the default configurations for model construction, which theoretically yield optimal path sets for efficient locality-preserving image traversal. More details of parameter selections, algorithm specifications, calculation time, etc., are provided in Appendix D.

4. Path Convolution Model

This section introduces the path convolution (PathConv) model, a CNN architecture with only 1D operators processing flattened 1D images. Figure 5 shows its overall architecture similar to canonical 2D CNNs (He et al., 2016; Liu et al., 2022). Input images are first transformed into 1D pixel streams by the path traversal layer, then fed into the stem layer, followed by PathConv blocks organized in four stages with downsampling layers in between. Besides, we propose a path-aware channel attention mechanism as an imperative component in our model to effectively process pixel segments from different regions of images. Detailed designs are elaborated in the following subsections.

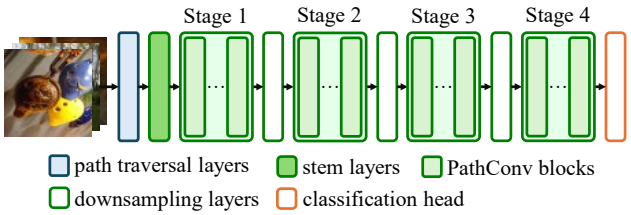


Figure 5. The overall architecture of PathConv models

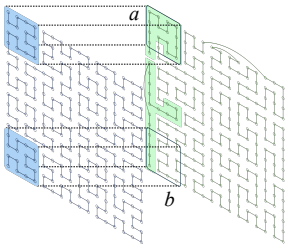


Figure 6. Given distinct paths (left: $\mathcal{H}_{16}$, right: $\mathcal{H}^c_{16}$, where $c = \{6, [0, 5], 0\}$), the model simultaneously observes (nearly) aligned regions in different orders (e.g., *Region a*: first 16 pixels), or disparate regions (e.g., *Region b*: last 16 pixels).

4.1. Path traversal layer

The path traversal layer flattens input images into 1D pixel streams following traversal orders specified by a set of paths consisting of sequential pixel coordinates. Note that this transformation is read-only, presenting significant potential for efficiency optimization. For example, we implement a CUDA kernel to perform fast image traversal by leveraging NVIDIA GPU parallelism, demonstrating up to 73-fold acceleration compared to a single-thread CPU implementation. We provide more details in Appendix E. We emphasize that the path traversal layer remains hardware-agnostic. Its overhead can become negligible through common optimization strategies, thus hardly downgrading the model efficiency.

The path traversal layer samples pixels along distinct paths. Hence, PathConv models simultaneously process pixel segments (a) from spatially aligned regions in varying orders or (b) from different spatial regions, as shown in Figure 6.

Both patterns are worth discussion as they fundamentally differ from conventional CNNs, generally processing the same spatial areas in the same sliding-window order.

Case (*a*) implies that 1D kernels handle pixels within (nearly) overlapping regions for multiple orders, which is similar to applying depthwise 1D convolution grouped by paths. This pattern also conceptually resembles oriented 1D convolution discussed in Section 2.1, where varying orientations (paths) enhance kernel expressiveness. Whereas case (*b*) necessitates the model to observe different regions of original images, effectively expanding the receptive fields. However, the corresponding kernel sizes remain the same, which may provide insufficient model capacity, potentially impeding model convergence.

4.2. Path-aware channel attention

To address this issue, we introduce a path-aware channel attention (PACA) module that captures both path-specific and cross-path dependencies, enabling models to focus dynamically on discriminative features. The parameter efficiency is maintained by using only channel-wise and path-wise attention, thereby avoiding excessive additional parameters. Given an input tensor $X \in \mathbb{R}^{B\times H\times L}$, where B is the batch size, H is the number of channels, and L is the input length, PACA processes features within and across paths.

We first organize X into path-specific groups to calculate intra-path attention, building upon efficient channel attention (Wang et al., 2020). For P paths and G channels per path, X is reshaped into $\hat{X} \in \mathbb{R}^{B\times P\times G\times L}$. We compute dual-scale attention weights for each path p using adaptive kernel sizes $k_1 = t + (1 - t \bmod 2)$ and $k_2 = 2k_1 - 1$, where $t = \lfloor \log_2 G/\gamma + \beta/\gamma \rfloor$, γ and β are hyperparameters controlling attention scales. We set $\gamma = 2$ and $\beta = 1$ in experiments. Then for each p, the corresponding X_p is averaged over L to obtain $y_p \in \mathbb{R}^{B\times 1\times G}$. Subsequently, the intra-path attention weights e_p are computed by

$$
\begin{aligned}
e_p^1 &= \texttt{Conv1D}(y_p, k_1) \\
e_p^2 &= \texttt{Conv1D}(y_p, k_2) \\
e_p &= \sigma(W_0[a_p^1; a_p^2] + b_0)
\end{aligned}
$$

where `Conv1D` are 1D convolutional layers for e_p^1 and e_p^2, $W_0 \in \mathbb{R}^{1\times 2}$ and $b_0 \in \mathbb{R}$ are learnable real-number parameters processing concatenated e_p^1 and e_p^2, and σ is the sigmoid activation function. The intra-path attention captures channel-wise dependencies at two scales, requiring merely $O(P \log G)$ parameters to obtain more robust channel scaling through dynamic balance between scales.

To model inter-path relationships, the concatenated path-specific e_p are firstly averaged across G to obtain $z \in \mathbb{R}^{B\times P}$. Then, the inter-path attention weights f are calculated through a multi-layer perception:

$$
f = \sigma(W_2\texttt{GELU}(W_1\texttt{LN}(z) + b_1) + b_2)
$$

where $W_1 \in \mathbb{R}^{P\times 2P}$, $b_1 \in \mathbb{R}^{2P}$, $W_2 \in \mathbb{R}^{2P\times P}$, $b_2 \in \mathbb{R}^P$ are parameters of two dense layers, `LN` denotes layer normalization (Ba et al., 2016), introducing $O(P^2)$ parameters in total. `GELU` serves as the activation function here (Hendrycks & Gimpel, 2016). The final attention weights w combine both intra- and inter-path attentions by $w = z \odot f$, where $\odot$ denotes element-wise multiplication with broadcasting. We can apply w to scale $\hat{X}$ across channels with global context relationships to produce the final output.

With appropriate feature scaling by PACA, we mitigate the convergence issue presented in case (*b*). Meanwhile, considering the number of paths P is constant (e.g., $|\mathbb{C}^*| = 3$), the entire PACA only introduces $O(\log G)$ additional parameters, growing logarithmically with model width, thus maintaining parameter efficiency. PACA modules are integrated into PathConv blocks introduced later.

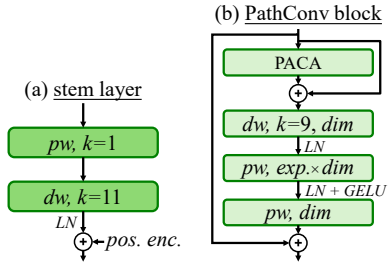


Figure 7. The stem layer and PathConv block designs.

4.3. Other building blocks

Stem design. Figure 7(a) illustrates the design of stem layers. A pointwise `Conv1D` first increases the channel dimension, followed by a depthwise `Conv1D` with a kernel size of 11. Compared to depthwise separable convolution (Chollet, 2017), reordering depthwise and pointwise `Conv1D` operations reduces the number of parameters. Additionally, learnable positional encoding (Dosovitskiy et al., 2021) enables the model to retain awareness of critical spatial information of pixels, with its necessity demonstrated by the ablation study (Table 5).

PathConv Blocks. The PathConv model utilizes PathConv blocks as the main building block, incorporating PACA modules before inverted bottlenecks (Sandler et al., 2018) with an expansion ratio of 4, as depicted in Figure 7(b). In the ablation study (Table 5), we validate that PACA is indispensable for PathConv models.

Others. The downsampling layers contain depthwise `Conv1D` with kernel sizes of 9 and strides of 2, followed by `LN` layers. The classification head is simply a dense layer transforming the dimensionality into the number of classes.

5. Experiments

We present experimental settings and results to evaluate the proposed PathConv model. Analysis of path selection demonstrates that $\mathbb{C}^*$ brings optimal performance. Ablation studies further validate the PathConv design.

5.1. Settings

Datasets. We evaluate PathConv models using three datasets: CIFAR-10 (Krizhevsky et al., 2009), SVHN (Netzer et al., 2011), and ImageNet-64 (Chrabaszcz et al., 2017), a downsampled version of ImageNet-1K (Russakovsky et al., 2015) with the same image count and dataset splits with a resolution of 64×64 across 1000 classes. We use low-resolution datasets as they amplify locality preservation issues (as shown in Table 1, P_{sd} grows for both paths as image resolution increases), allowing clear attribution of performance gains to our path-shifting method and design of components in PathConv models.

Models. PathConv models are constructed in two variants, small and base, denoted as PathConvS/B, with comparable floating-point operations (FLOPs) as ResNet-18/50 (He et al., 2016), respectively. The number of blocks (#bl.) in each stage and the corresponding number of channels (#ch.) are listed below:

- PathConvS: #bl.=(2, 2, 2, 2), #ch.=(48, 96, 192, 384)
- PathConvB: #bl.=(2, 2, 3, 3), #ch.=(60, 120, 240, 480)

We remove pooling layers from ResNet and change their large kernels into 3×3 to better adapt dataset resolutions. For ImageNet-64, the wide residual network (WRN) (Zagoruyko, 2016) is also considered, which is used in the paper introducing ImageNet-64 (Chrabaszcz et al., 2017).

Training. PathConv models employs the AdamW optimizer (Loshchilov & Hutter, 2019), while ResNet models utilize stochastic gradient descent with Nesterov momentum (Sutskever et al., 2013). All models are trained for 300 epochs with a 10-epoch linear warmup. The learning rate follows the cosine annealing schedule (Loshchilov & Hutter, 2017). We adopt RandAugment (Cubuk et al., 2020) for data augmentation. For ImageNet-64, we additionally apply Mixup (Zhang et al., 2018) and CutMix (Yun et al., 2019). Stochastic depth (Huang et al., 2016) and label smoothing (Szegedy et al., 2016) are also used for regularization. More experimental settings are provided in Appendix F.

5.2. Results

Table 3 shows the comparison results for three datasets regarding parameter counts (#param.), FLOPs and top-1 accuracy. PathConvS/B models incorporate suffixes denoting paths for the path sampling layers. Except for $\mathcal{H}_s^{\mathbb{C}^*}$ and $\mathcal{Z}_s^{\mathbb{C}^*}$ (see Table 2), $\mathcal{R}_s$ serves as a baseline for PathConv models.

Table 3. Performance comparison for three datasets. Numbers in parentheses show accuracy differences between PathConv variants and their ResNet counterparts with similar FLOPs. Green indicates improvement, red shows $\geq 2\%$ degradation, and amber denotes others. For WRN (WRN-*n*-*w*), *n* indicates the number of layers, and *w* denotes the width multiplier.

model	#param.	FLOPs	top-1 acc.
CIFAR-10 ($s = 32$)			
ResNet-18	11.68M	1.11G	93.17%
PathConvS-$\mathcal{H}_s^{\mathbb{C}^*}$	3.62M	1.14G	92.45% (-1.02%)
PathConvS-$\mathcal{Z}_s^{\mathbb{C}^*}$	3.62M	1.14G	92.66% (-0.51%)
PathConvS-$\mathcal{R}_s$	3.61M	1,14G	89.56% (-3.61%)
ResNet-50	25.55M	2.60G	93.95%
PathConvB-$\mathcal{H}_s^{\mathbb{C}^*}$	7.82M	2.50G	93.97% (+0.02%)
PathConvB-$\mathcal{Z}_s^{\mathbb{C}^*}$	7.82M	2.50G	93.56% (-0.39%)
PathConvB-$\mathcal{R}_s$	7.82M	2.49G	90.02% (-3.93%)
SVHN ($s = 32$)			
ResNet-18	11.68M	1.11G	97.66%
PathConvS-$\mathcal{H}_s^{\mathbb{C}^*}$	3.62M	1.14G	97.12% (-0.54%)
PathConvS-$\mathcal{Z}_s^{\mathbb{C}^*}$	3.62M	1.14G	96.57% (-1.09%)
PathConvS-$\mathcal{R}_s$	3.61M	1.14G	91.60% (-6.06%)
ResNet-50	25.55M	2.60G	97.86%
PathConvB-$\mathcal{H}_s^{\mathbb{C}^*}$	7.82M	2.50G	97.81% (-0.05%)
PathConvB-$\mathcal{Z}_s^{\mathbb{C}^*}$	7.82M	2.50G	97.58% (-0.28%)
PathConvB-$\mathcal{R}_s$	7.82M	2.49G	94.36% (-3.50%)
ImageNet-64 ($s = 64$)			
WRN-40-2	9.33M	3.83G	61.23%
ResNet-18	11.68M	4.44G	63.43%
PathConvS-$\mathcal{H}_s^{\mathbb{C}^*}$	3.76M	4.58G	62.26% (-1.17%)
PathConvS-$\mathcal{Z}_s^{\mathbb{C}^*}$	3.76M	4.58G	62.73% (-0.70%)
PathConvS-$\mathcal{R}_s$	3.76M	4.58G	58.04% (-5.39%)
WRN-40-5	57.24M	23.78G	70.32%
ResNet-50	25.55M	10.39G	68.43%
PathConvB-$\mathcal{H}_s^{\mathbb{C}^*}$	8.01M	9.98G	68.46% (+0.03%)
PathConvB-$\mathcal{Z}_s^{\mathbb{C}^*}$	8.01M	9.98G	68.83% (+0.40%)
PathConvB-$\mathcal{R}_s$	8.00M	9.97G	60.40% (-8.03%)

We can observe that PathConv models with $\mathcal{H}_s^{\mathbb{C}^*}$ and $\mathcal{Z}_s^{\mathbb{C}^*}$ consistently achieve comparable performance to ResNet while requiring <1/3 #param. and similar FLOPs. Notably, on ImageNet-64, PathConvS-$\mathcal{H}_s^{\mathbb{C}^*}/\mathcal{Z}_s^{\mathbb{C}^*}$ outperforms WRN-40-2 despite the latter's higher parameter count. PathConvB-$\mathcal{H}_s^{\mathbb{C}^*}/\mathcal{Z}_s^{\mathbb{C}^*}$ also perform better than ResNet-50. These results demonstrate the parameter efficiency of the proposed pure 1D PathConv architectures. Conversely, PathConv models utilizing $\mathcal{R}_s$ have significantly lower performance than ResNet, indicating that locality-preserving paths generated by our path-shifting method ($\mathbb{C}^*$) are more effective for image traversal in 1D PathConv models than raster scan paths, corroborating the analysis presented in Section 3 and also illustrating that PathConv models effectively leverage the locality-preserving properties of $\mathcal{H}_s^{\mathbb{C}^*}$ and $\mathcal{Z}_s^{\mathbb{C}^*}$.

Table 4. Performance comparison of PathConvB models using varying image traversal paths. Numbers in parentheses indicate accuracy differences of $\mathcal{H}_s^{\mathbb{C}^*}/\mathcal{Z}_s^{\mathbb{C}^*}$ from $\mathcal{H}_s^{\mathbb{C}^+}/\mathcal{Z}_s^{\mathbb{C}^+}$ and $\mathcal{H}_s/\mathcal{Z}_s$. The same color schema as Table 3 is applied here.

model	#param.	FLOPs	top-1 acc.
CIFAR-10 ($s = 32$)			
PathConvB-$\mathcal{H}_s^{\mathbb{C}^*}$	7.82M	2.50G	93.97%
PathConvB-$\mathcal{H}_s^{\mathbb{C}^+}$	7.82M	2.50G	91.01% (-2.96%)
PathConvB-$\mathcal{H}_s$	7.82M	2.49G	94.12% (+0.15%)
PathConvB-$\mathcal{Z}_s^{\mathbb{C}^*}$	7.82M	2.50G	93.56%
PathConvB-$\mathcal{Z}_s^{\mathbb{C}^+}$	7.82M	2.50G	90.98% (-2.58%)
PathConvB-$\mathcal{Z}_s$	7.82M	2.49G	93.23% (-0.33%)
SVHN ($s = 32$)			
PathConvB-$\mathcal{H}_s^{\mathbb{C}^*}$	7.82M	2.50G	97.81%
PathConvB-$\mathcal{H}_s^{\mathbb{C}^+}$	7.82M	2.50G	96.57% (-1.24%)
PathConvB-$\mathcal{H}_s$	7.82M	2.49G	97.23% (-0.58%)
PathConvB-$\mathcal{Z}_s^{\mathbb{C}^*}$	7.82M	2.50G	97.58%
PathConvB-$\mathcal{Z}_s^{\mathbb{C}^+}$	7.82M	2.50G	97.04% (-0.44%)
PathConvB-$\mathcal{Z}_s$	7.82M	2.49G	97.31% (-0.27%)
ImageNet-64 ($s = 64$)			
PathConvB-$\mathcal{H}_s^{\mathbb{C}^*}$	8.01M	9.98G	68.46%
PathConvB-$\mathcal{H}_s^{\mathbb{C}^+}$	8.01M	9.98G	62.23% (-6.23%)
PathConvB-$\mathcal{H}_s$	8.00M	9.97G	60.53% (-7.93%)
PathConvB-$\mathcal{Z}_s^{\mathbb{C}^*}$	8.01M	9.98G	68.83%
PathConvB-$\mathcal{Z}_s^{\mathbb{C}^+}$	8.01M	9.98G	63.07% (-5.76%)
PathConvB-$\mathcal{Z}_s$	8.00M	9.97G	60.01% (-8.82%)

5.3. The impact of path selection

We now examine the influence of path selection on PathConv model performance. The results in Table 3 demonstrate negligible performance variations between $\mathcal{H}_s^{\mathbb{C}^*}$ and $\mathcal{Z}_s^{\mathbb{C}^*}$. This invariance to path type indicates the robustness of our path-shifting method, as both paths effectively satisfy the locality constraint, enabling PathConv models to achieve performance comparable to ResNet models.

Beyond path types, we also investigate whether $|\mathbb{C}^*| = 3$ is optimal for PathConv models. To this end, $\mathbb{C}^+$ is defined as a set of c presented in Table 10 (Appendix F) with doubled cardinality (i.e., $P = 6$) satisfying the locality constraint. Meanwhile, the original $\mathcal{H}_s$ and $\mathcal{Z}_s$ are also path candidates for $P = 1$. We apply these paths to PathConvB models for three datasets, with results shown in Table 4.

For small datasets, $P = 1$ brings limited adverse effects, occasionally providing better performance (CIFAR-10) than $\mathbb{C}^*$, while $P = 6$ results in more severe performance drops. These findings indicate that PathConv models achieve better convergence with fewer paths on small datasets, enabling better results even without satisfying the locality constraint. Meanwhile, despite PACA, $P = 6$ exacerbates the problem of the case (b), impeding model convergence. However, $P = 1$ performs significantly worse than $\mathbb{C}^*$ and $\mathbb{C}^+$ for PathConvB, as it fails to meet the locality constraint, thus providing insufficient model capacity for large datasets (ImageNet-64). In conclusion, $|\mathbb{C}^*| = 3$ perfectly balances model efficiency and performance. These insights empirically suggest $\mathbb{C}^*$ as the optimal set for PathConv models.

Table 5. Results for PathConvB-$\mathcal{H}_s^{\mathbb{C}^*}$ on three datasets following the ablation of positional encoding and PACA components, with the coloring scheme in Table 3.

pos. enc.	PACA	top-1 acc.
CIFAR-10 ($s = 32$)		
✓	✓	93.97%
✓	✗	90.13% (-3.84%)
✗	✓	92.34% (-1.63%)
✗	✗	86.23% (-7.74%)
SVHN ($s = 32$)		
✓	✓	97.12%
✓	✗	94.20% (-2.92%)
✗	✓	95.91% (-1.21%)
✗	✗	89.77% (-7.35%)
ImageNet-64 ($s = 64$)		
✓	✓	68.46%
✓	✗	60.93% (-7.53%)
✗	✓	63.85% (-4.61%)
✗	✗	52.91% (-15.55%)

5.4. Ablation study

We validate that both positional encoding and PACA are essential for PathConv models by ablation. Table 5 presents the corresponding results of removing either or both components from PathConvB-$\mathcal{H}_s^{\mathbb{C}^*}$. These results indicate that removing either module substantially degrades model performance, with PACA's removal having a more pronounced effect. The absence of PACA significantly impairs the ability of PathConv models to process cases (b). Given the small #param. requirements for both modules, the architecture of PathConv models has proven effective and efficient.

6. Conclusion

This paper presents a novel approach to constructing CNNs using exclusively 1D operations, achieving parameter efficiency while maintaining performance comparable to 2D CNNs. We introduce a path-shifting method to effectively preserve the locality of flattened 2D images, requiring only three paths to outperform trivial raster scan paths. Building upon this foundation, our PathConv architecture, featuring path-aware channel attention, matches the performance of ResNet while using only 1/3 of the parameters across multiple datasets. This work reveals a new paradigm for efficient CNN design based purely on 1D operations.

Acknowledgements

This work was supported by the Federal Ministry of Education and Research of Germany (BMBF) under grant number 16DHBKI020.

Impact Statement

This paper presents work that aims to advance the field of computer vision. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Ansari, A. and Fineberg, A. Image data ordering and compression using peano scan and lot. *IEEE Transactions on Consumer Electronics*, 38(3):436–445, 1992.

Ba, L. J., Kiros, J. R., and Hinton, G. E. Layer normalization. *CoRR*, abs/1607.06450, 2016. URL http://arxiv.org/abs/1607.06450.

Bertsimas, D. and Vohra, R. Rounding algorithms for covering problems. *Mathematical Programming*, 80:63–89, 1998.

Böhm, C., Perdacher, M., and Plant, C. A novel hilbert curve for cache-locality preserving loops. *IEEE Transactions on Big Data*, 7(2):241–254, 2018.

Castro, J., Georgiopoulos, M., Demara, R., and Gonzalez, A. Data-partitioning using the hilbert space filling curves: Effect on the speed of convergence of fuzzy artmap for large database problems. *Neural Networks*, 18(7):967–984, 2005.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2018. doi: 10.1109/TPAMI.2017.2699184.

Chen, W., Zhu, X., Chen, G., and Yu, B. Efficient point cloud analysis using hilbert curve. In *European Conference on Computer Vision*, pp. 730–747. Springer, 2022.

Chen, W., Yao, X., Zhang, X., and Yu, B. Efficient deep space filling curve. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 17525–17534, 2023.

Cheng, C. and Parhi, K. K. Fast 2d convolution algorithms for convolutional neural networks. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 67(5):1678–1691, 2020.

Chollet, F. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1251–1258, 2017.

Chrabaszcz, P., Loshchilov, I., and Hutter, F. A downsampled variant of imagenet as an alternative to the cifar datasets. *arXiv preprint arXiv:1707.08819*, 2017.

Cohen, T. and Welling, M. Group equivariant convolutional networks. In *International Conference on Machine Learning*, pp. 2990–2999. PMLR, 2016.

Cubuk, E. D., Zoph, B., Shlens, J., and Le, Q. V. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020.

Dai, H. and Su, H.-C. On the locality properties of space-filling curves. In *International Symposium on Algorithms and Computation*, pp. 385–394. Springer, 2003.

Denton, E. L., Zaremba, W., Bruna, J., LeCun, Y., and Fergus, R. Exploiting linear structure within convolutional networks for efficient evaluation. *Advances in Neural Information Processing Systems*, 27, 2014.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

Feige, U. A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)*, 45(4):634–652, 1998.

Freeman, W. and Adelson, E. The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(9):891–906, 1991. doi: 10.1109/34.93808.

Freeman, W. T., Adelson, E. H., et al. The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(9):891–906, 1991.

Fukushima, K. Neocognitron: A hierarchical neural network capable of visual pattern recognition. *Neural Networks*, 1(2):119–130, 1988. ISSN 0893-6080. doi: https://doi.org/10.1016/0893-6080(88)90014-7.

Girshick, R. Fast r-cnn. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1440–1448, 2015. doi: 10.1109/ICCV.2015.169.

Guo, M.-H., Lu, C.-Z., Hou, Q., Liu, Z., Cheng, M.-M., and Hu, S.-M. Segnext: Rethinking convolutional attention design for semantic segmentation. *Advances in Neural Information Processing Systems*, 35:1140–1156, 2022.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

Hendrycks, D. and Gimpel, K. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Hilbert, D. *Über die stetige Abbildung einer Linie auf ein Flächenstück*, pp. 1–2. Springer Berlin Heidelberg, Berlin, Heidelberg, 1935. ISBN 978-3-662-38452-7. doi: 10.1007/978-3-662-38452-7_1. URL https://doi.org/10.1007/978-3-662-38452-7_1.

Huang, G., Sun, Y., Liu, Z., Sedra, D., and Weinberger, K. Q. Deep networks with stochastic depth. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pp. 646–661. Springer, 2016.

Huang, T., Yin, L., Zhang, Z., Shen, L., Fang, M., Pechenizkiy, M., Wang, Z., and Liu, S. Are large kernels better teachers than transformers for convnets? In *International Conference on Machine Learning*, pp. 14023–14038. PMLR, 2023.

Jaderberg, M., Vedaldi, A., and Zisserman, A. Speeding up convolutional neural networks with low rank expansions. In *BMVC 2014-Proceedings of the British Machine Vision Conference 2014*. British Machine Vision Association, 2014.

Jagadish, H. V. Linear clustering of objects with multiple attributes. In *Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data*, pp. 332–342, 1990.

Jin, J., Dundar, A., and Culurciello, E. Flattened convolutional neural networks for feedforward acceleration. *CoRR*, abs/1412.5474, 2014.

Kamel, I. and Faloutsos, C. Hilbert r-tree: An improved rtree using fractals. In *VLDB*, volume 94, pp. 500–509. Citeseer, 1994.

Kirchmeyer, A. and Deng, J. Convolutional networks with oriented 1d kernels. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6222–6232, 2023.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, May 2017. ISSN 0001-0782. doi: 10.1145/3065386. URL https://doi.org/10.1145/3065386.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989. doi: 10.1162/neco.1989.1.4.541.

Lempel, A. and Ziv, J. Compression of two-dimensional data. *IEEE Transactions on Information Theory*, 32(1): 2–8, 1986.

Li, G., Shen, X., Li, J., and Wang, J. Diagonal-kernel convolutional neural networks for image classification. *Digital Signal Processing*, 108:102898, 2021.

Lin, S., Ji, R., Chen, C., Tao, D., and Luo, J. Holistic cnn compression via low-rank decomposition with knowledge transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(12):2889–2905, 2018.

Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., and Xie, S. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11976–11986, 2022.

Loshchilov, I. and Hutter, F. SGDR: stochastic gradient descent with warm restarts. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019.

Morton, G. M. A computer oriented geodetic data base and a new technique in file sequencing. 1966.

Mukherjee, D. and Mukhopadhyay, S. Fast hardware architecture for 2-d separable convolution operations. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 65(12):2042–2046, 2018.

Narayanan, D., Phanishayee, A., Shi, K., Chen, X., and Zaharia, M. Memory-efficient pipeline-parallel dnn training. In *International Conference on Machine Learning*, pp. 7937–7947. PMLR, 2021.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A. Y., et al. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, pp. 4. Granada, 2011. URL http://ufldl.stanford.edu/housenumbers.

Peano, G. Sur une courbe, qui remplit toute une aire plane. *Mathematische Annalen*, 36:157–160, 1890. doi: 10.1007/BF01199438.

Peng, C., Zhang, X., Yu, G., Luo, G., and Sun, J. Large kernel matters–improve semantic segmentation by global convolutional network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4353–4361, 2017.

Raghavan, P. and Tompson, C. D. Randomized rounding: a technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–374, 1987.

Ravanbakhsh, S., Oliva, J., Fromenteau, S., Price, L., Ho, S., Schneider, J., and Póczos, B. Estimating cosmological parameters from the dark matter distribution. In *International Conference on Machine Learning*, pp. 2407–2416. PMLR, 2016.

Rigamonti, R., Sironi, A., Lepetit, V., and Fua, P. Learning separable filters. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2754–2761, 2013.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, 2018.

Sutskever, I., Martens, J., Dahl, G., and Hinton, G. On the importance of initialization and momentum in deep learning. In *International Conference on Machine Learning*, pp. 1139–1147. PMLR, 2013.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818–2826, 2016.

Szegedy, C., Ioffe, S., Vanhoucke, V., and Alemi, A. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

Vazirani, V. V. *Set Cover*, pp. 15–26. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003. ISBN 978-3-662-04565-7. doi: 10.1007/978-3-662-04565-7_2.

Wang, H., Gupta, K., Davis, L., and Shrivastava, A. Neural space-filling curves. In *European Conference on Computer Vision*, pp. 418–434. Springer, 2022.

Wang, Q., Wu, B., Zhu, P., Li, P., Zuo, W., and Hu, Q. Eca-net: Efficient channel attention for deep convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

Weiler, M., Hamprecht, F. A., and Storath, M. Learning steerable filters for rotation equivariant cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 849–858, 2018.

Worrall, D. E., Garbin, S. J., Turmukhambetov, D., and Brostow, G. J. Harmonic networks: Deep translation and rotation equivariance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5028–5037, 2017.

Wu, X., Jiang, L., Wang, P.-S., Liu, Z., Liu, X., Qiao, Y., Ouyang, W., He, T., and Zhao, H. Point transformer v3: Simpler faster stronger. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4840–4851, 2024.

Yang, H., Tang, M., Wen, W., Yan, F., Hu, D., Li, A., Li, H., and Chen, Y. Learning low-rank deep neural networks via singular vector orthogonality regularization and singular value sparsification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition workshops*, pp. 678–679, 2020.

Yoo, A. B., Jette, M. A., and Grondona, M. Slurm: Simple linux utility for resource management. In *Workshop on job scheduling strategies for parallel processing*, pp. 44–60. Springer, 2003.

Yun, S., Han, D., Oh, S. J., Chun, S., Choe, J., and Yoo, Y. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6023–6032, 2019.

Zagoruyko, S. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

Zhang, H., Cissé, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.

Červený, J. Generalized hilbert (gilbert) space-filling curve. https://github.com/jakubcerveny/gilbert, 2024.

A. Pixel-wise comparison of locality preservation for multiple resolutions

We extend pixel-wise comparison of locality preservation with the same comparison method and coloring schema as Figure 3 to multiple resolutions. Figure 8 illustrates pixel-wise locality preservation status for Hilbert/Z-order paths compared to raster scan paths of the same sizes. We can observe that the shape of purple pixels also roughly grows with topological self-organizing properties, aligning with the recursive definition of both paths.

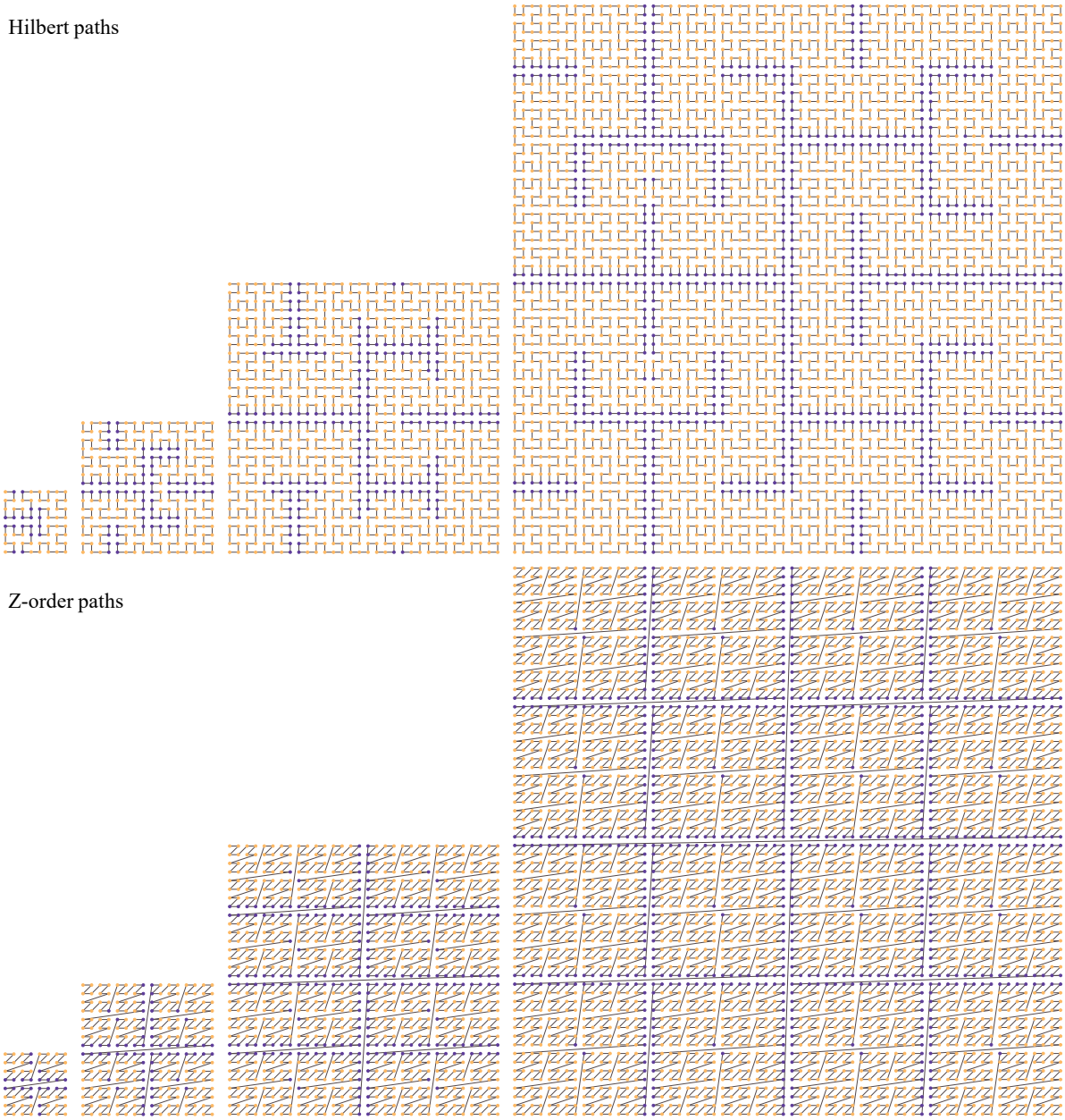
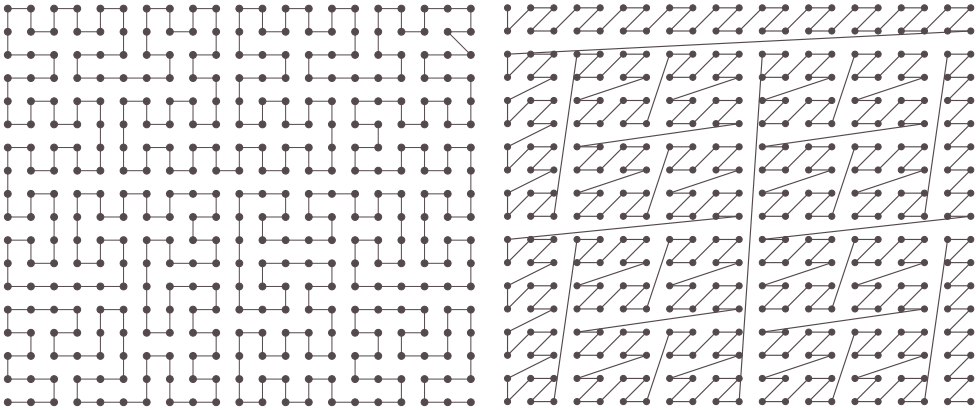


Figure 8. Applying the same comparison method and coloring schema utilized in Figure 3 with 8^2, 16^2, 32^2, 64^2 resolutions. Pixels with shorter cumulative distances to all their neighbors than raster scan paths are in **apricot**, otherwise **purple**.

B. Supports for rectangular images

Except for $\mathcal{R}_s$ inherently accommodating various resolutions, $\mathcal{H}_s$ also supports rectangular images through minor modifications, such as generalized Hilbert curves (*gilbert*) (Červený, 2024). Figure 9 (a) shows an example of 18×21 generalized

Hilbert path using *gilbert*. In comparison, original $\mathcal{Z}_s$ only accepts squares with $s \in \{2^n \mid n \in \mathbb{N}\}$, where $\mathbb{N}$ is the set of natural numbers. The proposed path-shifting method manages to fill this gap, as shown in Figure 9 (b). We obtain this 18×21 Z-order path by sampling from $\mathcal{Z}_{32}$ with $i = 0, j = 5, r = 0$. Hence, our shifting method also unintentionally extends the application scope of $\mathcal{Z}_s$ for arbitrary image sizes. Consequently, our method could be applied to any image dataset without compatibility constraints.



(a) 18×21 generalized Hilbert path (b) 18×21 Z-order path by path shifting

Figure 9. 18×21 paths as examples of paths for rectangular sizes.

C. Reduction to the set cover problem

We first formally describe our problem of determining $\mathbb{C}$ with the smallest possible cardinality to meet the locality constraint and the set cover problem, then show the reduction procedure.

PROBLEM 1 (P1) *Minimal $\mathbb{C}$ with the locality constraint.* Let $\mathcal{C} = \{c_1, ..., c_n\}$ be the universe of all possible configurations given s, each configuration c_q is associated with

- a binary matrix $M_q \in \{0, 1\}^{s \times s}$ representing pixel-wise locality preservation status as shown in Figure 4(d). If a pixel at $[j, k]$ has shorter distances to its neighbors, $M_q[j, k] = 1$, otherwise $M_q[j, k] = 0$.
- a distance measurement $d_q \in \mathbb{N}$ recording the total distance of this path (the same as Table 1).

Determine a subset $\mathbb{C} \subseteq \mathcal{C}$ minimizing

$$\sum\{d_q \mid c_q \in \mathbb{C}\} \tag{1}$$

subject to the locality constraint

$$\sum\{M_q[j, k] \mid c_q \in \mathbb{C}\} \geq 1, \forall j, k \in [0, s) \tag{2}$$

We formulate this problem as an optimization procedure without loss of generality as minimizing (1) inherently prevents involving unnecessary paths.

PROBLEM 2 (P2) *Weighted Set Cover Problem.* Given a universe $\mathcal{U}$, a family of sets $\mathcal{F} = \{F_1, ..., F_r\}$, where each set $F \subseteq \mathcal{U}$, and a cost function $w : F \to \mathbb{N}$. Find a subset $\mathbb{F} \subseteq \mathcal{F}$ minimizing

$$\sum\{w(F) \mid F \in \mathbb{F}\} \tag{3}$$

subject to

$$\bigcup\{F \mid F \in \mathbb{F}\} = \mathcal{U} \tag{4}$$

Reduction. Consider the following reduction η mapping any instance of P1 to an instance of P2 as follows:

- Let $\mathcal{U} = \{[j, k] \mid \forall j, k \in [0, s)\}$, where each element is a pixel position in P1.

- For each configuration $c_q \in \mathcal{C}$, create a set $F_q = \{(x, y) \mid M_q[x, y] = 1]\}$. Then, $\mathcal{F} = \{F_q \mid i \in [1, r]\}$.
- For every F_q corresponding to c_q, let $w(F_q) = d_q$.

Following this mapping, the reduction η respectively suffices to these three properties:

- η is polynomial-time computable, as only $O(ns^2)$ operations are required to construct $\mathcal{U}$ and $\mathcal{F}$ from an instance of P1, where n is the number of configurations and s^2 is the size of images.
- η preserves solution mapping. This is because for any $\mathbb{C}$ of an instance of P1, the set $\{F_q \mid c_q \in \mathbb{C}\}$ is also a solution to the corresponding instance of P2, as any pixel (j, k) with $M_q[j, k] = 1$ corresponds to (j, k) covered by F_q, and vice versa.
- η maintains solution optimality by construction. For any $\mathbb{C}$ of an instance of P1 and its corresponding solution $\mathbb{F}$, $\sum\{d_q \mid c_q \in \mathbb{C}\} = \sum\{w(F) \mid F \in \mathbb{F}\}$.

Therefore, P1 is polynomial-time reducible to P2. Existing algorithms and theoretical bounds of the set cover problem could be used to find and evaluate satisfactory $\mathbb{C}$ in our problem.

D. More details of obtaining $\mathbb{C}^*$

In the previous section, we have presented detailed steps to map our problem of finding $\mathbb{C}^*$ to an instance of the weighted set cover problem. This formulation enables us to employ the randomized rounding algorithm (Raghavan & Tompson, 1987), a widely adopted approach for solving set cover problems.

Let x_q be a binary variable indicating whether configuration c_q is selected. We reuse definitions in Appendix C. Then, finding $\mathbb{C}^*$ can be formulated as the following integer linear program (ILP):

minimize

$$\sum\{d_q x_q \mid c_q \in \mathcal{C}\}$$

subject to the locality constraint

$$\sum\{M_q[j, k]\ x_q \mid c_q \in \mathcal{C}\} \geq 1, \forall j, k \in [0, s),$$

where $x_q \in \{0, 1\}$. The randomized rounding algorithm begins by relaxing the constraint $x_q \in 0, 1$ to $0 \leq x_q \leq 1, x_q \in \mathbb{R}$, thereby transforming this ILP into a linear program (LP) solvable in polynomial time. After obtaining the optimal fractional solution $\hat{x}$, the algorithm executes Φ-round iterations. During each iteration, it selects c_q with probability $\min(\alpha\hat{x_q}, 1)$, where α denotes a scaling factor. When the selected configuration set satisfies the locality constraint (2), the algorithm updates the optimal solution if a shorter total distance is achieved. Our implementation utilizes parameters $\Phi = 30$ and $\alpha = 1.2$. While employed in this study, the randomized rounding algorithm is not our contribution. Detailed algorithmic analysis is available in (Raghavan & Tompson, 1987).

Table 6. Comparison of the randomized rounding and greedy algorithms on finding $\mathbb{C}^*$, showing average computational time and the cardinality of the worst-case solution $|\mathbb{C}^*|$. Greedy algorithm cannot always guarantee $|\mathbb{C}^*| = 3$.

s	Path	$\lvert\mathcal{C}\rvert$	Randomized rounding algorithm		Greedy algorithm	
			avg. time	worst case $\lvert\mathbb{C}^*\rvert$	avg. time	worst case $\lvert\mathbb{C}^*\rvert$
32	Hilbert	39684	67.1s	3	8.3s	3
	Z-order	3969	2.0s	3	0.1s	3
64	Hilbert	333316	7134.6s	3	634.6s	4
	Z-order	16129	321.7s	3	51.9s	4

While the randomized rounding algorithm provides a theoretical approximation ratio of $O(\ln s^2)$ with high probability (Feige, 1998), its computational cost significantly exceeds that of the trivial greedy algorithm (Table 6). This motivates an investigation into whether the greedy algorithm can achieve comparable solution quality with reduced computational

Table 7. Runtime comparison between our CUDA kernel for image traversal and its CPU counterpart for varying batch sizes, s, and the number of paths P. Each measurement represents the average execution time over 1,000 iterations, recorded in milliseconds (ms).

batch size	s	$P = 3$		$P = 5$	
		CPU	Ours	CPU	Ours
128	32	0.755ms	0.083ms (9.112x)	0.830ms	0.076ms (10.909x)
	64	1.648ms	0.075ms (22.065x)	2.400ms	0.094ms (25.489x)
	224	25.920ms	0.369ms (70.250x)	33.891ms	0.555ms (61.026x)
512	32	1.506ms	0.077ms (19.521x)	2.141ms	0.093ms (22.933x)
	64	8.074ms	0.158ms (51.181x)	10.070ms	0.212ms (47.393x)
	224	80.864ms	1.319ms (61.287x)	123.716ms	2.091ms (59.166x)
1024	32	4.083ms	0.105ms (38.860x)	5.297ms	0.133ms (39.962x)
	64	15.877ms	0.253ms (62.823x)	21.508ms	0.366ms (58.690x)
	224	190.583ms	2.587ms (73.659x)	254.165ms	4.160ms (61.101x)

Table 8. Training settings of PathConv models for three datasets.

Setting	CIFAR-10	SVHN	ImageNet-64
initial learning rate	4e-3	4e-3	1e-2
weight decay	2e-4	2e-4	2e-4
optimizer	AdamW	AdamW	AdamW
optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.999$	$\beta_1, \beta_2 = 0.9, 0.999$	$\beta_1, \beta_2 = 0.9, 0.999$
batch size	512	512	384
scheduler	cosine annealing	cosine annealing	cosine annealing
warmup epoch	10	10	10
warmup schedule	linear	linear	linear
RandAugment	7	7	9
Mixup	-	-	0.2
CutMix	-	-	0.6
label smoothing	0.05	0.05	0.05

overhead. We implement the greedy solver by iteratively picking c that covers the most pixels with longer distances to their neighbors than raster scanning. Table 6 provides the comparison results of using randomized rounding and greedy algorithms to find $\mathbb{C}^*$. The average computational time is calculated across three executions of each algorithm. Despite the significantly shorter computational time of the greedy solver, it does not consistently guarantee $|\mathbb{C}^*| = 3$. Consequently, the randomized rounding algorithm remains the preferred choice, particularly given that $\mathbb{C}^*$ computation is required only once.

E. CUDA implementation of the path sampling layer and efficiency comparison

We implemented a specialized CUDA kernel to accelerate the image traversal procedure and integrated it into the path sampling layer for maximum efficiency. The kernel assigns one thread per pixel position to compute its corresponding index in the resulting 1D pixel stream. We specify 256 threads grouped together for every CUDA block, enabling shared GPU memory access, cooperation, and synchronization among threads within each block. Table 7 compares runtimes between our CUDA kernel implementation and CPU version for varying batch sizes, s, and the number of paths P on 1 NVIDIA A100 GPU. Our CUDA kernel achieves up to 73.659-fold acceleration compared to the CPU implementation, demonstrating the efficiency of the path sampling layer.

We emphasize that the image traversal procedure is read-only, enabling optimization through various parallel processing approaches. Our CUDA implementation here is one example, selected here due to its direct benefits for model training.

Table 9. Training settings of ResNet/WRN models for three datasets.

Setting	CIFAR-10	SVHN	ImageNet-64
initial learning rate	2e-2	2e-2	2e-2
weight decay	2e-4	2e-4	2e-4
optimizer	SGD	SGD	SGD
optimizer momentum	0.9	0.9	0.9
batch size	512	512	384
scheduler	cosine annealing	cosine annealing	cosine annealing
warmup epoch	10	10	10
warmup schedule	linear	linear	linear
RandAugment	7	7	9
Mixup	-	-	0.2
CutMix	-	-	0.6
label smoothing	0.05	0.05	0.05

Table 10. The configurations denoted as $\mathbb{C}^+$ satisfying the locality constraint when $|\mathbb{C}^+| = 6$ for Hilbert and Z-order paths. We specify $\mathbb{C}^+$ for Section 5.3 to investigate the impact of path selection.

Path	s	Configurations ($\mathbb{C}^+$)
Hilbert	32/64	{ 0, [0, 0], 0}, { 5, [0, 4], 0}, { 0, [4, 0], 0}, { 0, [4, 4], 0}, { 0, [2, 2], 0}, { 0, [2, 4], 0}
Z-order	32	{ 0, [0, 0], 0}, {32, [16, 26], 0}, {32, [14, 16], 180}, {32, [12, 20], 180}, {32, [8, 4], 0}, {32, [4, 8], 0}
	64	{ 0, [0, 0], 0}, {64, [36, 32], 270}, {64, [36, 34], 90}, {64, [28, 32], 0}, {64, [4, 8], 0}, {64, [8, 4], 0}

F. Experimental settings

We provide detailed training settings for PathConv models and 2D CNNs in Table 8 and Table 9, respectively. We do not apply weight decay on all normalization layers (`LN`). To ensure a fair comparison, all models are trained from scratch. This eliminates confounding factors from transfer learning and allows direct attribution of performance differences to architectural choices.

To investigate the impact of path selection, we also specify $\mathbb{C}^+$ with $|\mathbb{C}^+| = 6$ for Section 5.3 as given in Table 10.