Efficient Deep Space Filling Curve

Wanli Chen, Xufeng Yao, Xinyun Zhang, Bei Yu
The Chinese University of Hong Kong
{wlchen,xfyao,xyzhang21,byu}@cse.cuhk.edu.hk

We provide details of learning scheme, discussion on the scalability of proposed method and more experimental results in appendix. Code will be released publicly.

A. Details of Learning Scheme

The algorithm of training our model is described as follow:

Algorithm 1 Training

Initialize main network $\mathrm{Conv}, \mathrm{L}, \mathrm{EGCN}$;
Initialize siamese network $\mathrm{Conv}^s, \mathrm{L}^s, \mathrm{EGCN}^s$;
for number of training epochs **do**
  Select a mini-batch B images;
  for every image $\mathfrak{I}$ in the B **do**
    Generate random noise $N \sim \mathcal{N}(0, 1)$;
    $\boldsymbol{W}_{\mathcal{G}'} \leftarrow \mathrm{EGCN}(\mathrm{L}(\mathrm{Conv}(\mathfrak{I})))$;
    $\boldsymbol{W}^s_{\mathcal{G}'} \leftarrow \mathrm{EGCN}^s(\mathrm{L}^s(\mathrm{Conv}^s(\mathfrak{I} + N)))$;
    Obtain SFCs via $\mathcal{T}$ and Cover_and_Merge;
    Calculate objectives Φ^s, Φ^m;
  end for
  Calculate average $\bar{\Phi}^s$ using Φ^s;
  Calculate average $\bar{\Phi}^m$ using Φ^m;
  if $\bar{\Phi}^s$ is better **then**
    Fix $\mathrm{Conv}^s, \mathrm{L}^s, \mathrm{EGCN}^s$;
    Loss function $\mathcal{L} \leftarrow \mathcal{L}_{KD}$
    Update $\mathrm{Conv}, \mathrm{L}, \mathrm{EGCN}$ using Adam optimizer
  else
    Fix $\mathrm{Conv}, \mathrm{L}, \mathrm{EGCN}$
    Loss function $\mathcal{L} \leftarrow \mathcal{L}_{KD}$
    Update $\mathrm{Conv}^s, \mathrm{L}^s, \mathrm{EGCN}^s$ using Adam optimizer
  end if
end for

B. Scalability of Our Method

In the following experiments, we verify the scalablity of proposed algorithm. We first ablate the GPU occupation of EGCN with different size of input grid graphs. Then, we conduct experiments on Tiny-imagenet dataset with various input resolutions.

Method	Size	GPU Memory
GCN [4]	128×128	1248M
EGCN	128×128	**128M**
GCN [4]	256×256	17280M
EGCN	256×256	**513M**

Table 1: All methods are tested by using grid graphs with batch size 512. With the increase of grid size, the GPU cannot afford the memory cost produced by GCN.

Dataset	Method	Autocorrelation↑
Tiny-imagenet (128×128) [5]	Zigzag	0.799 (-0.072)
	Hilbert [2]	0.871
	Ours	**0.881 (+0.01)**
Tiny-imagenet (256×256) [5]	Zigzag	0.768 (-0.154)
	Hilbert [2]	0.922
	Ours	**0.929 (+0.007)**

Table 2: Zigzag curve is implemented by Pytorch reshape function. The autocorrelation factor $k = 100$ for experiments.

Scalability of EGCN. In this experiment, we ablate grid graph with different sizes and test the GPU memory occupation to evaluate the efficiency of different GCN methods. The experimental setting is identical to the main paper, which adopts 3-layer GCN networks for measurement and the tested GPU is RTX 3090 Ti.

The results in Table 1 show that our proposed EGCN consumes much lower GPU memory compared with GCN method, which illustrates that our EGCN is scalable to large image.

Scalability of Proposed Framework. We also show the scalability of our framework by increasing the input image size to 256×256 with Tiny-imagenet. The results in Table 2 demonstrate that our model is scalable to different sizes of the image. However, the performance improvement is marginally decreased with the increasing image resolution. The potential reason is that our learning scheme may not find the global optimal MST. Searching for an optimal MST

Method	Ensemble	Autocorrelation	Training Time	Inference Time
2-predictor	×	0.625	T_{train}	T_{test}
2-predictor	✓	0.626	T_{train}	$2T_{test}$
3-predictor	×	0.625	$1.5T_{train}$	T_{test}
3-predictor	✓	0.628	$1.5T_{train}$	$3T_{test}$
4-predictor	×	0.626	$2T_{train}$	T_{test}
4-predictor	✓	0.629	$2T_{train}$	$4T_{test}$

Table 3: Ablation on multi-teacher strategy. T_{train} and T_{test} stand for the training and testing time of the main paper setting.

on large images is a difficult problem and we will continue finding a better learning scheme to solve the problem in the future work.

C. More Ablation Studies

We follow the same ablation setting as the main paper, which uses the MNIST dataset with the autocorrelation objective (k=6).

Ablation on Multiple Teacher and Model Ensemble. Our learning scheme can be viewed as a two-predictor structure and we select the predictor with higher performance as the teacher while regarding the other as the student. It is natural to extend this scheme to a multiple-predictor structure. Therefore, in this experiment, we add multiple networks that share the same setting with the siamese network that mentioned in our paper. For example, the input for all networks are original images with noise ($\mathcal{I} + \mathcal{N}(0, 1)$). Then, we set the best predictor as the teacher while others as the student. The parameters of students are updated by measuring the KL-divergence with the output features of the teacher network. The original setting in the main paper is denoted as 2-predictor and we propose N-predictor ($N \geq 2$). The experimental results can be found in Table 3.

Here ensemble indicates for choosing the best prediction as the final output, which is different from the setting in our main paper that only use the main network for inference. It shows that the performance improvement introduced by the multiple-predictor structure and model ensemble are trivial. Considering the time consumption, these operations are not worthy.

Compared with random search. In the optimization phase, we incorporated a siamese network with noisy input to enhance the output diversity. To demonstrate the superiority of our learning-based approach over random search, we conducted experiments. First, we generated random MST and applied the Cover-and-Merge algorithm to create a random SFC (referred to as B_1-SFC) as the first baseline. Next, we used an untrained network to perform direct inference, creating another random SFC (referred to as B_2-SFC) as the second baseline. In Table 5, we compared the autocorrelation performance of these SFC search methods. The MNIST experimental results clearly indicate that our learning scheme

Method	Train Params	Test Params	TGIF(LZW)	FFHQ(AC)	FFHQ(LZW)
NSFC	22.9M	11.7M	556.9	0.911	678.3
Ours	**22.3M**	**11.1M**	**552.4**	**0.925**	**671.9**

Table 4: Comparison with NSFC. AC means autocorrelation ($k = 10$) and LZW means LZW code length.

Learning Scheme	Autocorrelation
B_1-SFC	0.376
B_2-SFC	0.382
Ours	**0.625**

Table 5: Comparison between our scheme and others.

Noise Type	Autocorrelation
$U(-1, 1)$	0.624
$\mathcal{N}(0, 1)$	0.625

Table 6: Comparison between two noise types.

Method	Dict Length↓	Scan Order Array Size (MB)
Hilbert [2]	437.72	0.0156
Dafner [1]	419.81	0.0156
NSFC [7]	413.65	0.0156
Ours	**407.28**	0.0156

Table 7: Average LZW dictionary length of different methods. Scan order array is an 2×1024 array which stores the scan order of SFC.

significantly outperforms random search.

Noise Selection. In our main paper, we added random Gaussian noise $\mathcal{N}(0, 1)$ to the inputs of the siamese network to increase input diversity. However, uniform noise $U(-1, 1)$ is also a viable option. Therefore, we conducted an ablation study to determine the optimal noise type. The experimental results are presented in Table 6, which shows that the choice of noise has a negligible impact on the results. This indicates that the model performance is not determined by the type of noise used, as long as it can introduce input diversity to the siamese network.

D. More Comparison

More datasets. In Table 4 we compare the performance of our method with NSFC [7] on the TGIF [6] and FFHQ [3] datasets. Unlike NSFC, which generates SFC per-class, our method generates SFC per-image. This means that the generated curve is adapted to each image, resulting in better quality in terms of LZW code length and autocorrelation.

Dictionary length. In addition to the LZW coding results, we test the dictionary length of the LZW code that can be used to decode compressed sequences. We use the MNIST dataset during the experiment, and a shorter length indicates a more repetitive pattern, which implies better clustering properties since homogeneous areas are clustered. The results in Table 7 show that our generated SFC requires the smallest average LZW dictionary length. Although our method achieves better results than NSFC on LZW benchmarks, our advantage lies in generating SFCs for individual images. Since NSFC [7] generates SFCs for each class, it can amortize the memory overhead of storing the LZW dictionary and SFC scan order. In contrast, our method generates an SFC for each image, which, although yielding better performance, incurs a larger memory overhead compared to NSFC in multi-image tasks such as video compression (In our method, storing the scan order array for 10K images will cost 156MB).

References

[1] Revital Dafner, Daniel Cohen-Or, and Yossi Matias. Context-based space filling curves. In *Computer Graphics Forum*, volume 19, pages 209–218. Wiley Online Library, 2000. 2

[2] David Hilbert. Über die stetige abbildung einer linie auf ein flächenstück. In *Dritter Band: Analysis· Grundlagen der Mathematik· Physik Verschiedenes*, pages 1–2. Springer, 1935. 1, 2

[3] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019. 2

[4] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016. 1

[5] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015. 1

[6] Dian Qin, Haishuai Wang, Zhe Liu, Hongjia Xu, Sheng Zhou, and Jiajun Bu. Hilbert distillation for cross-dimensionality networks. *arXiv preprint arXiv:2211.04031*, 2022. 2

[7] Hanyu Wang, Kamal Gupta, Larry Davis, and Abhinav Shrivastava. Neural space-filling curves. *arXiv preprint arXiv:2204.08453*, 2022. 2, 3