Π-nets: Deep Polynomial Neural Networks

Grigorios G. Chrysos[1], Stylianos Moschoglou[1,2], Giorgos Bouritsas[1],
Yannis Panagakis[3], Jiankang Deng[1,2], Stefanos Zafeiriou[1,2]
[1] Imperial College London, [2] Facesoft.io, [3] University of Athens Greece

Imperial College London

Imperial College London

Motivation

- Impressive results have been obtained using deep convolutional neural networks the last few years. Skip connections have been extensively used for further improvements of such networks.
- However, even with such powerful networks, image generation is largely not solved. The current state-of-the-art networks [4, 2] use some special kind of skip connections.
- Motivated by the impressive performance by such networks, we demonstrate how such special skip connections create high-order polynomials.
- Our new class of neural networks, called Π−nets, has strong empirical results in a battery of tasks.

Method

- Given input $\boldsymbol{z}$, we want to approximate the function $G(\boldsymbol{z})$.
- The classic feedforward networks, use the form: $\boldsymbol{x}_n = \boldsymbol{S}_{[n]}^T \boldsymbol{x}_{n-1} + \boldsymbol{b}_{[n]}$ where n indicates the n^{th} layer for $n \in [1, N]$. An activation function is typically applied after the $\boldsymbol{x}_n$ before the next layer.
- Instead, we want to use an alternative approximator, i.e. polynomials. We define the recursive form:

$$\boldsymbol{x}_n = \left(\boldsymbol{A}_{[n]}^T \boldsymbol{z}\right) * \left(\boldsymbol{S}_{[n]}^T \boldsymbol{x}_{n-1} + \boldsymbol{B}_{[n]}^T \boldsymbol{b}_{[n]}\right) \quad (1)$$

- The symbol '$*$' refers to an elementwise product.
- Instead of using a single polynomial, we can use a product of polynomials, i.e. the output of a polynomial (1) can be the input for the next polynomial. That enables an exponential total order.

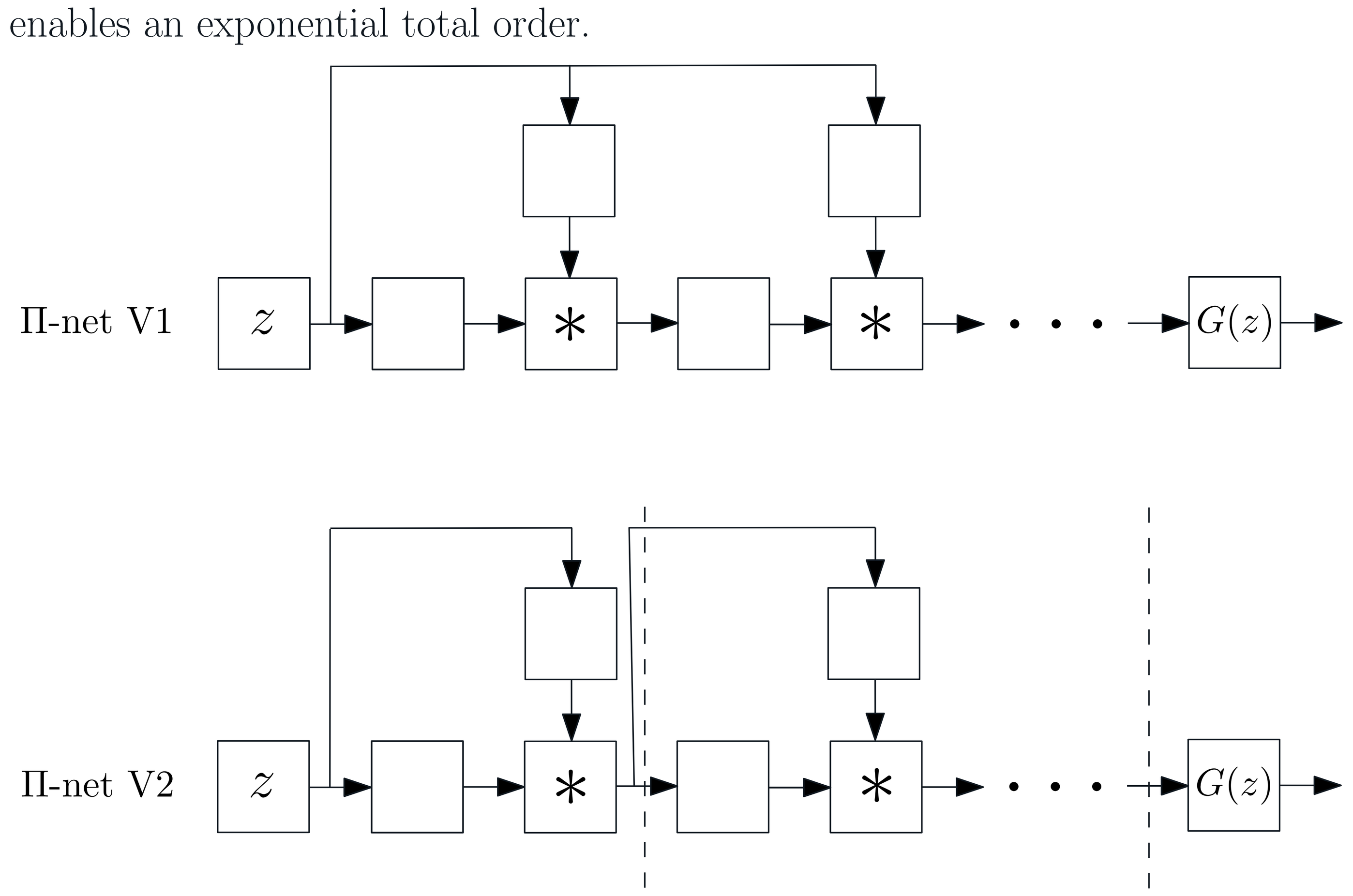

Fig. 1: Two schematics of Π−nets. The one on the top expresses a single polynomial, while the one on the bottom illustrates a product of polynomials.

Generation/classification without activation functions

- We train polynomial generators with linear blocks, i.e. ditching the activation functions between the layers, in a GAN setting:

Fig. 2: Linear interpolation in the latent space of a polynomial generator when trained on a) fashion images [8], b) colored MNIST, c) facial images [3].

- Noticeably, the generator without activation functions between the layers can learn the data distributions.
- We perform a similar experiment with a polynomial classification network, i.e. we remove the activation functions of the network. The accuracy in CIFAR-10 and CIFAR-100 is **90.7%** and **66.7%** respectively.

Experiments against state-of-the-art methods

We conduct experiments against state-of-the-art methods in a) face generation, b) audio classification.

- We modify the state-of-the-art StyleGAN [4] and convert it into a product of polynomials; the performance improves by **4.6%**. Synthesized faces from our method:

- In the second experiment we use the Speech Commands dataset. The goal is to demonstrate the increased expressivity of Π−nets. The accuracy of the compared methods (in the Table below) is similar, but Π−net has 38% fewer parameters. The symbol '# par' abbreviates the number of parameters (in millions).

Speech Commands classification			
Model	# blocks	# par (↓)	Accuracy (↑)
ResNet34	$[3, 4, 6, 3]$	21.3	0.951 ± 0.002
Π−net	$[3, 3, 3, 2]$	**13.2**	0.951 ± 0.002

3D mesh representation learning

We conduct an experiment on 3D deformable meshes of fixed topology. We extend the state-of-the-art spiral convolutions [1] by converting them into a polynomial:

	error (mm) (↓)	speed (ms) (↓)
GAT [6]	0.732	11.04
FeastNet [7]	0.623	6.64
MoNet [5]	0.583	7.59
SpiralGNN [1]	0.635	4.27
Π−net (simple)	0.530	4.98
Π−net (simple - linear)	0.529	4.79
Π−net (full)	0.476	5.30
Π−net (full - linear)	**0.474**	5.14

Aside of the quantitative results, we provide qualitative visualization:

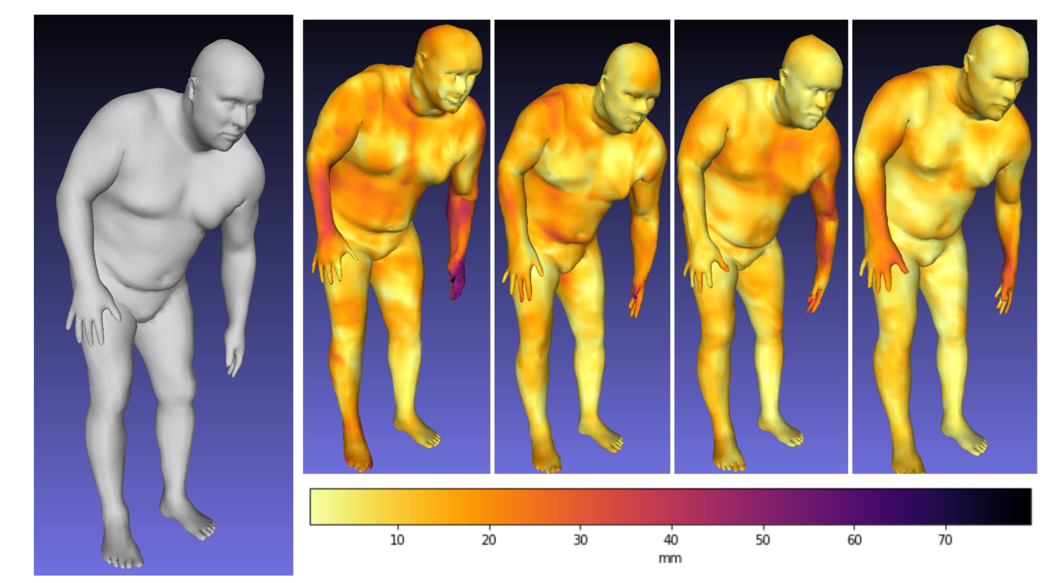

Fig. 4: Color coding of the per vertex reconstruction error on an exemplary human body mesh. From left to right: ground truth mesh, 1st order SpiralGNN, 2^{nd}, 3^{rd} and 4^{th} order Π−net.

References

[1] Giorgos Bouritsas et al. "Neural 3D Morphable Models: Spiral Convolutional Networks for 3D Shape Representation Learning and Generation". In: *ICCV*. 2019.

[2] Andrew Brock, Jeff Donahue, and Karen Simonyan. "Large scale gan training for high fidelity natural image synthesis". In: *ICLR*. 2019.

[3] Athinodoros S Georghiades, Peter N Belhumeur, and David J Kriegman. "From few to many: Illumination cone models for face recognition under variable lighting and pose". In: *T-PAMI* 6 (2001), pp. 643–660.

[4] Tero Karras, Samuli Laine, and Timo Aila. "A style-based generator architecture for generative adversarial networks". In: *CVPR*. 2019.

[5] Federico Monti et al. "Geometric deep learning on graphs and manifolds using mixture model cnns". In: *CVPR*. 2017.

[6] Petar Veličković et al. "Graph Attention Networks". In: *ICLR* (2018).

[7] Nitika Verma, Edmond Boyer, and Jakob Verbeek. "Feastnet: Feature-steered graph convolutions for 3d shape analysis". In: *CVPR*. 2018.

[8] Han Xiao, Kashif Rasul, and Roland Vollgraf. "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms". In: *arXiv preprint arXiv:1708.07747* (2017).