Hourglass MLP: Rethinking the Shape of Residual Architectures

Meng-Hsi Chen[1]* † **Yu-Ang Lee**[1,2]* **Feng-Ting Liao**[1]† **Da-shan Shiu**[1]

[1]**MediaTek Research** [2]**National Taiwan University**

Abstract

Multi-layer perceptrons (MLPs) conventionally adopt a narrow–wide–narrow design, where residual connections operate at input/output dimensions and computation occurs in expanded hidden spaces. We revisit this convention with **wide–narrow–wide (Hourglass)** MLP blocks, where residual connections act at the widest stage while computation flows through narrow bottlenecks. This inversion leverages expressive high-dimensional residual spaces for stable feature refinement. A key enabler is an initial expansion projection that can remain fixed at random initialization, reducing training and inference costs. We evaluate Hourglass MLPs on generative tasks and extend the design to Vision Transformers (ViTs) across multiple scales. Experiments demonstrate consistently superior performance–parameter Pareto frontiers compared to conventional designs. The reduced bottleneck cost enables flexible parameter reallocation toward wider residual representations or increased depth under fixed budgets. Our findings establish skip connection placement as a critical design principle for compute-optimal scaling in residual architectures.

1. Introduction

Multi-layer perceptrons (MLPs) traditionally follow a "narrow–wide–narrow" shape, where feature expansion occurs in hidden layers while residual connections operate at narrower dimensions [1, 2]. We challenge this convention, hypothesizing that *incremental refinement is more effective at higher dimensionality*. Theoretically, this is supported by *Cover's Theorem* and the *Johnson-Lindenstrauss Lemma*, which suggest that lifting data to higher-dimensional spaces increases linear separability and preserves geometric structures, providing a more expressive landscape for additive residual corrections[3, 4]. Unlike models like LoRA [5] which use wide–narrow–wide shapes to *approximate* updates, our Hourglass design utilizes this structure as a fundamental principle to *enhance* representation refinement.

We propose the *wide–narrow–wide (Hourglass)* architecture, which inverts the standard MLP to ensure residual paths operate at the network's widest stage (Figure 1(a)). This design enables compute-optimal scaling: parameters saved from narrow bottlenecks can be reallocated to expand the latent dimension d_z or network depth L. For pure MLPs, we lift inputs via a projection W_{in}, which we show can remain fixed at random initialization, facilitating efficient exploration of expanded residual spaces without additional training costs.

Our main contributions are:

- **Architectural Inversion:** We introduce the Hourglass MLP, performing residual updates in a high-dimensional latent space to leverage superior learning dynamics in wide representations.

* These authors contributed equally.

† Correspondence: meng-hsi.chen@mtkresearch.com, ft.liao@mtkresearch.com

- **Efficiency on Pure MLPs:** Systematic searches show Hourglass consistently achieves superior PSNR-parameter Pareto frontiers in vision generative tasks, even with fixed random input projections.
- **Scalability in ViTs:** We extend this principle to Vision Transformers [6], where Hourglass variants consistently outperform baselines across diverse scales, yielding a 3.8% gain on Stanford Cars [7] with ViT-Tiny and maintaining robust improvements as the model scales up.

2. Wide–narrow–wide Incremental–improving Architectures

Grounded in *Cover's theorem* [3] and *compressive sensing* [8], we propose inverting the conventional narrow–wide–narrow design into wide–narrow–wide (Hourglass) blocks. We hypothesize that skip connections operating at higher dimensions enable more effective incremental refinement. This design allows saving parameters from narrow bottlenecks to be reallocated toward wider latent spaces or increased depth under a fixed budget, a strategy we systematically evaluate through Pareto frontiers in Section 3.2.

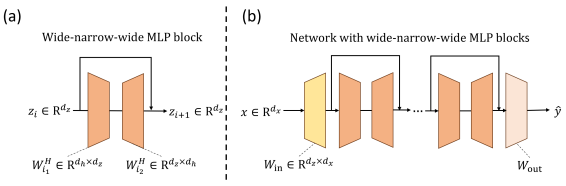


Figure 1: (a) Wide-narrow-wide MLP block. Residual connections link high-dimensional endpoints z_i, z_{i+1} ($d_z > d_h$), inverting the conventional low-dimensional skip placement. Normalization and nonlinearities are omitted for clarity. (b) Full network architecture. A stack of Hourglass blocks using an input projection W_{in} to lift x into latent space z , and an output projection W_{out} for task adaptation.

2.1. The Hourglass Residual Block

The core innovation lies in performing the residual update within a high-dimensional space. For each block i, the computation follows:

$$z_{i+1} = z_i + W^H_{i_2}\, \sigma_i\big(W^H_{i_1}\, \text{norm}(z_i)\big)\,, \tag{1}$$

where $W^H_{i_1} \in \mathbb{R}^{d_h \times d_z}$ and $W^H_{i_2} \in \mathbb{R}^{d_z \times d_h}$. By ensuring $d_z > d_h$, the identity path preserves information in a richer linear space, mitigating "rank collapse" as demonstrated by our representation analysis in Section 3.2.3. Unlike LoRA [5], which uses this shape for low-rank approximation, our design utilizes it as a fundamental principle for representation enhancement.

2.2. Pure Hourglass MLP Networks

A pure Hourglass MLP network comprises three stages (Figure 1(b)):

Input-to-latent projection. The signal $x \in \mathbb{R}^{d_x}$ is lifted via $z_0 = W_{\text{in}}x$, where $d_z > d_x$. Drawing from the *Johnson-Lindenstrauss lemma* [4] and *reservoir computing* [9], we propose fixing W_{in} at its random initialization. This reduces training costs while maintaining comparable performance to learned projections, as empirically verified in Section 3.2.2.

Processing and Output. The latent representation flows through L blocks and is converted by W_{out} to the task format (e.g., $\hat{y} = \text{softmax}(W_{\text{out}}z_L)$ for classification). This structure allows the network to learn optimal output latents before task-specific finetuning.

2.3. Extension to Transformer Architectures

We replace the conventional $1 \to 4 \to 1$ FFN in Vision Transformers (ViTs) with our wide–narrow–wide FFN (Figure 2(b)). The resulting parameter efficiency enables increasing the base model dimension d_z or total layers L. Consequently, both the Multi-Head Attention (MHA) and FFN operate within the same expanded residual space, fostering more expressive relational modeling. Our scaling experiments in Section 3.3 confirm that this reallocation yields superior performance across multiple scales.

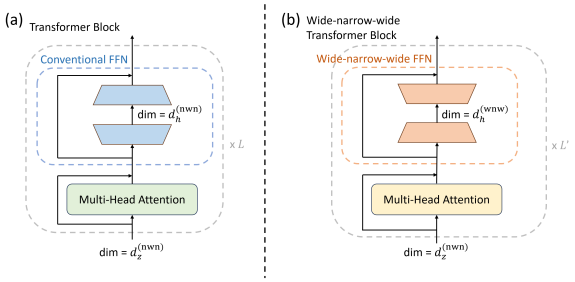


Figure 2: (a) The classic transformer block with Multi-Head Attention (MHA) and a conventional narrow–wide–narrow FFN. (b) A modified transformer block with the proposed wide–narrow–wide Hourglass FFN, where the dimensionality of MHA is adjusted accordingly.

2.4. Shape and Parameter Reallocation

The Hourglass paradigm decouples residual space (d_z) from computation width (d_h). Our findings favor a "deep-and-wide-latent" regime: maximizing representational capacity via wider skip connections while maintaining efficiency through narrow bottlenecks. This scaling behavior, distinct from conventional designs, is consistently observed in both our MLP and ViT results in Section 3.

3. Experiments and Results

We first benchmark Hourglass MLPs against conventional baselines on generative tasks to evaluate Pareto frontiers, scaling trajectories (d_z, d_h, L), and fixed W_{in} viability. We further extend this principle to ViTs to study resource reallocation from narrow FFN bottlenecks into depth or attention width, characterizing architectural efficiency and generality across scales.

3.1. Experimental Setup

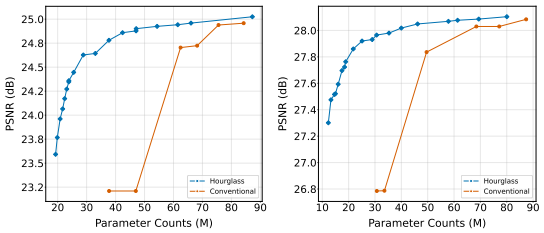
Hourglass MLP. Tasks include denoising, super-resolution, and generative classification on MNIST [10], ImageNet-32 [11], and ImageNet-224 [12] to assess incremental signal refinement. We isolate block shape as the primary variable under identical training protocols. Search parameters cover d_z, d_h, L, and W_{in} strategies. Main results focus on ImageNet-224 (Section 3.2); other details are in Appendix A and C.

Hourglass ViT. We use the training-from-scratch protocol in Nakamura et al. [13] across CIFAR-10/100 [14], Stanford Cars [7], Oxford Flowers-102 [15], and ImageNet-100 [12, 13]. Models span Tiny, Small, and Large scales. Architectural parameters are in Table 2; dataset and training details are deferred to Appendix D.

3.2. Main Results for Hourglass MLP

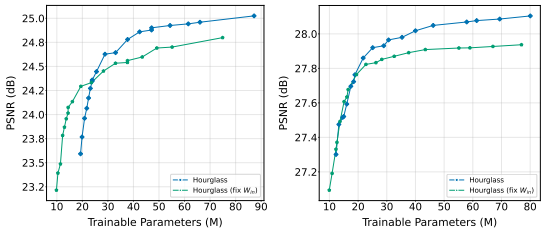
3.2.1. GENERATIVE RESTORATION TASKS

On ImageNet-224 (Figure 3), Hourglass MLPs consistently outperform conventional baselines in denoising and super-resolution. In denoising (Figure 3(a)), Hourglass achieves $24.959\,\text{dB}$ with 66M parameters, surpassing the 84M-parameter conventional model. In super-resolution (Figure 3(b)), Hourglass attains $28.086\,\text{dB}$ with 69M parameters, while the best conventional model requires 87M.



(a) Denoising (b) Super-resolution

Figure 3: Restoration on ImageNet-224. Pareto fronts on denoising and super-resolution. Optimal configurations are shown in Appendix B.



(a) Denoising (b) Super-resolution

Figure 4: Fixed W_{in} results. Minor degradation while pushing the frontier toward the upper-left.

The advantage is most pronounced in mid-range budgets , suggesting that high-dimensional residual updates significantly enhance restoration fidelity and efficiency.

3.2.2. EFFECT OF FIXED VS. TRAINABLE INPUT PROJECTION

To verify if random projections preserve signal information, we compare fixed versus trainable W_{in} in Hourglass models. Figure 4 shows that fixed W_{in} maintains competitive Pareto frontiers, sometimes shifting them toward the upper-left. This indicates that gains from learning W_{in} are minor, offering a parameter-efficient alternative for resource-constrained settings.

3.2.3. REPRESENTATION ANALYSIS VIA EFFECTIVE RANK

Table 1: Effective rank analysis. Despite its increased depth, the Hourglass architecture achieves higher PSNR and effective rank. This indicates a more expressive utilization of the high-dimensional latent space. Analyzed on Pareto-optimal ImageNet-32 super-resolution configurations (Table 9).

Architecture	**Params (M)**	d_z	d_h	L	**PSNR**	**Effective Rank**
Conventional	55.37	3072	3546	2	23.8783	1180.63
Hourglass	55.00	3075	1146	6	**23.9701**	**1501.27**

To counter rank collapse [16], Hourglass trades width d_h for depth L, achieving a 27% higher effective rank (1501.27 vs. 1180.63). This improved high-dimensional space utilization correlates with a 0.37 dB PSNR gain, theoretically justifying the architecture's superior expressivity [17].

3.3. Main Results for Hourglass ViT

3.3.1. CONVENTIONAL VS. HOURGLASS VIT

Table 3 presents the results for ViT-Tiny. Our Hourglass variant consistently outperforms the conventional baseline by 3.0% on average across four datasets. Notably, on fine-grained tasks such as Stanford Cars and Oxford Flowers, the performance gain reaches +4.0% and +5.0%, respectively. This confirms that for an equivalent parameter budget, the Hourglass design's ability to operate MHA in a higher-dimensional space ($d_z = 288$ vs. 192) significantly boosts representational capacity.

3.3.2. SCALABILITY ACROSS MODEL SCALES

Table 4 details scaling experiments. The Hourglass architecture maintains a consistent lead as models scale. On ViT-Large, our Hourglass model achieves **21.3%** on Stanford Cars, surpassing

Table 2: ViT model configurations. Hourglass variants widen default d_z by $1.5\times$ while contracting the MLP ratio (d_h/d_z) to $0.5\times$. Depths L are adjusted to match parameter counts and training FLOPs of conventional baselines. Parameters include the 196-label CARS classification head.

Model	Architecture	d_z	d_h	L	Heads	Params (M)	Training Cost (GFLOPs)	Inference Latency (ms/image)
ViT-Tiny	Conventional	192	768	12	3	5.56	7.46	0.2456
	Hourglass	288	144	11	3	4.93	7.04	0.2224
ViT-Small	Conventional	384	1536	12	6	21.74	27.48	0.7636
	Hourglass	576	288	12	6	20.64	27.10	0.7923
ViT-Large	Conventional	1024	4096	24	16	303.50	369.02	10.0496
	Hourglass	1536	768	25	16	297.07	367.40	9.9432

Table 3: ViT-Tiny performance across six benchmarks. *Conventional*[*] denotes values from Nakamura et al. [13]; *Conventional* indicates our optimized training. Hourglass architectures, trained from scratch, show consistent advantages across tasks. See Table 2 for configurations and Appendix E.1 for training curves.

Architecture	CIFAR-10	CIFAR-100	Cars	Flowers	ImageNet-100	Average
Conventional[*]	78.3	57.7	11.6	77.1	73.2	59,6
Conventional	82.3	66.1	12.3	77.3	79.5	63.5
Hourglass	**84.1**	**66.8**	**16.1**	**81.5**	**80.6**	**65.9**

the conventional baseline while utilizing fewer parameters (296M vs. 303M) and an additional layer ($L = 25$). Regarding Oxford Flowers training dynamics, Hourglass-Large exhibits faster convergence than Small and Tiny variants (Figure 13), suggesting more efficient optimization in higher-dimensional residual spaces.

Table 4: Scale comparison of conventional and Hourglass ViTs. Trained from scratch, Hourglass shows consistent gains across diverse backbone sizes. See Appendix E.2 for training curves.

Dataset	ViT-Tiny		ViT-Small		ViT-Large	
	Conventional	Hourglass	Conventional	Hourglass	Conventional	Hourglass
Cars	12.3	**16.1**	15.9	**18.6**	19.0	**21.3**
Flowers	77.3	**81.5**	79.7	**81.9**	81.2	**82.6**

4. Discussion and Future Work

Our findings show that high-dimensional residual updates enhance refinement and capacity utilization. While vision-proven up to ViT-Large, future work should explore other modalities and sensitivities to normalization or activation. To scale the Hourglass principle, we propose: (1) stacking N iterative FFN refinement blocks; (2) searching for optimal (d_z, d_h, L) ratios in Transformers; and (3) integrating MHLA to decouple complexity from d_z. This synergy establishes Hourglass architectures as a foundation for compute-optimal scaling.

References

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017. URL https://arxiv.org/abs/1706.03762.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision (ECCV)*, pages 630–645. Springer, 2016. URL https://arxiv.org/abs/1603.05027.

[3] Thomas M. Cover. Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE Transactions on Electronic Computers*, EC-14(3): 326–334, 1965. doi: 10.1109/PGEC.1965.264137.

[4] William B Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. In *Conference in modern analysis and probability*, volume 26 of *Contemporary Mathematics*, pages 189–206. American Mathematical Society, 1984.

[5] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=nZe72R8yS0.

[6] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[7] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 554–561, 2013.

[8] David L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4): 1289–1306, 2006. doi: 10.1109/TIT.2006.871582.

[9] Herbert Jaeger. The "echo state" approach to analysing and training recurrent neural networks. Technical report, German National Research Center for Information Technology, 2001.

[10] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

[11] Patryk Chrabaszcz, Ilya Loshchilov, and Frank Hutter. A downsampled variant of imagenet as an alternative to the CIFAR datasets. *CoRR*, abs/1707.08819, 2017. URL http://arxiv.org/abs/1707.08819.

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[13] Ryo Nakamura, Hirokatsu Kataoka, Sora Takashima, Edgar Josafat Martinez Noriega, Rio Yokota, and Nakamasa Inoue. Pre-training vision transformers with very limited synthesized images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20360–20369, 2023.

[14] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[15] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, pages 722–729. IEEE, 2008.

[16] Ben Poole, Subhaneil Lahiri, Maithra Raghu, Jascha Sohl-Dickstein, and Surya Ganguli. Exponential expressivity in deep neural networks through transient chaos. *Advances in neural information processing systems*, 29, 2016.

[17] Hadi Daneshmand, Jonas Kohler, Francis Bach, Thomas Hofmann, and Aurelien Lucchi. Batch normalization provably avoids ranks collapse for randomly initialised deep networks. *Advances in Neural Information Processing Systems*, 33:18387–18398, 2020.

[18] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[19] Ross Wightman. PyTorch Image Models. URL https://github.com/huggingface/pytorch-image-models.

[20] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

Contents

Appendix A. Detailed Experimental Setup for Hourglass MLP

A.1. Datasets and Tasks

Table 5 summarizes the datasets, tasks, and input/output signal dimensions of our experimentation on pure MLPs. Specifically for ImageNet, to avoid MLP parameter explosion, we apply custom data preprocessing to the original ImageNet dataset to reduce its input size while maintaining high-resolution information. As shown in Figure 5, we first follow the standard procedure to resize each image to $256 \times 256 \times 3$ and center crop to $224 \times 224 \times 3$. Next, we select the center $96 \times 96 \times 3$ region in each image to avoid background and other low-information pixels. Finally, the selected $96 \times 96 \times 3$ region is further cropped into 9 patches of $32 \times 32 \times 3$ images, resulting in 11.5M training images and 225K validation and test images. This procedure results in the same input size as ImageNet-32, facilitating MLP training while retaining much higher-resolution information in the data.

Dataset	Task	Input Size	Output Size	Description
MNIST	Generative Classification	$28 \times 28 \times 1$	$28 \times 28 \times 1$	Generate GT image for predicted class
MNIST	Denoising	$28 \times 28 \times 1$ (noisy)	$28 \times 28 \times 1$	Remove artificially added noise
MNIST	Super-resolution	$14 \times 14 \times 1$	$28 \times 28 \times 1$	Recover high-resolution handwritten image
ImageNet-32	Denoising	$32 \times 32 \times 3$ (noisy)	$32 \times 32 \times 3$	Remove artificially added noise
ImageNet-32	Super-resolution	$16 \times 16 \times 3$	$32 \times 32 \times 3$	Recover high-resolution natural scene image
ImageNet-224	Denoising	$32 \times 32 \times 3$ (noisy)	$32 \times 32 \times 3$	Remove artificially added noise
ImageNet-224	Super-resolution	$16 \times 16 \times 3$	$32 \times 32 \times 3$	Recover high-resolution natural scene image

Table 5: Summary of datasets, tasks, and input/output sizes

(a) Original image ($900 \times 585 \times 3$) (b) Resized and cropped ($224 \times 224 \times 3$) (c) Final cropped ($32 \times 32 \times 3$)

Figure 5: **Example of our custom data preprocessing on ImageNet.** Unlike ImageNet-32, which directly downsamples the original image to 32×32, the approach here retains higher-resolution pixel information in the final 32×32 patches.

A.2. Model Configuration Search Ranges and Training Setups

All experiments were conducted using NVIDIA RTX A6000 and RTX 3090 GPUs. The images were mapped to [0,1] before training, and we employed the AdamW [18] optimizer with a linear learning rate scheduler and no warm-up period.

MNIST. The original training set of 60,000 images was randomly partitioned into 50,000 samples for training and 10,000 for validation, while the original test set of 10,000 images was reserved for final evaluation. The MLP architectural parameters were searched over the ranges $d_h \in [4, 2500]$,

$d_z \in [785, 4500]$, and $L \in [1, 40]$, while the learning rate $\in \{1\times10^{-4}, 5\times10^{-4}, 1\times10^{-3}, 5\times10^{-3}\}$. . All experiments were repeated 5 times, and we report the mean and standard deviation ($\mu \pm \sigma$) across runs. Note that during grid search, we constrained $d_z > d_x$ and $d_h < d_z$ for the Hourglass architecture, while $d_h > d_z$ for the conventional MLP, following their respective architectural definitions.

- **Generative Classification:** Ground truth images were randomly selected for each digit. Training was conducted with a batch size of 128 for 50 epochs.
- **Denoising:** Noisy images were prepared by adding Gaussian noise (mean = 0, std = 0.25). Training used batch size 128 for 50 epochs.
- **Super-resolution:** Downscaled images were prepared using bicubic interpolation, reducing the original $28 \times 28 \times 1$ images to $14 \times 14 \times 1$. Training applied $4\times$ data augmentation (original, horizontal flip, vertical flip, and combined horizontal-vertical flip) with batch size 128 for 50 epochs.

ImageNet-32. The complete original training set of 1,281,167 images was utilized for training, and the original validation set of 50,000 images was randomly split into 25,000 samples for validation and 25,000 for testing. We report the performance on the test set using the model that achieved the lowest validation loss. The MLP architectural parameters were searched over the ranges $d_h \in [4, 2500]$, $d_z \in [3072, 4576]$, and $L \in [1, 30]$, while the learning rate $\in \{1\times10^{-4}, 3\times10^{-4}, 5\times10^{-4}, 7\times10^{-4}\}$. All experiments were repeated 5 times, and we report the mean and $5\times$ standard deviation ($\mu \pm 5\sigma$) across runs. Note that during grid search, we constrained $d_z > d_x$ and $d_h < d_z$ for the Hourglass architecture, while $d_h > d_z$ for the conventional MLP, following their respective architectural definitions.

- **Denoising:** Noisy images were prepared by adding Gaussian noise (mean = 0, std = 0.25). Training used $4\times$ data augmentation (original, horizontal flip, vertical flip, and combined horizontal-vertical flip) with batch size 512 for 2 epochs.
- **Super-resolution:** Downscaled images were prepared using bicubic interpolation, reducing the original $32 \times 32 \times 3$ images to $16 \times 16 \times 3$. Training applied $4\times$ data augmentation (original, horizontal flip, vertical flip, and combined horizontal-vertical flip) with batch size 512 for 2 epochs.

ImageNet-224. The preprocessed ImageNet dataset contains 11.5M training images and 225K validation and test images. As with ImageNet-32, the MLP architectural parameters were searched over the ranges $d_h \in [4, 2500]$, $d_z \in [3072, 4576]$, and $L \in [1, 30]$, while the learning rate was searched over $\{1 \times 10^{-4}, 3 \times 10^{-4}, 5 \times 10^{-4}, 7 \times 10^{-4}\}$. All experiments were repeated 5 times, and we report the mean and 5 times the standard deviation ($\mu \pm 5\sigma$) across runs.

- **Denoising:** Noisy images were prepared by adding Gaussian noise (mean = 0, std = 0.25). Training used batch size 512 for 1 epoch.
- **Super-resolution:** Downscaled images were prepared using bicubic interpolation, reducing the original $32 \times 32 \times 3$ images to $16 \times 16 \times 3$. Training used batch size 512 for 1 epoch.

Appendix B. Pareto-optimal Configurations for Hourglass MLP

Across denoising and super-resolution tasks on ImageNet-224, Table 6 and Table 7 reveal three consistent trends: (1) **Superior Efficiency:** Hourglass models consistently achieve higher PSNR while using substantially fewer parameters; (2) **Structural Preference:** Optimal Hourglass configurations favor deeper networks ($L \geq 4$) with moderate bottlenecks ($d_h \in [115, 765]$), whereas conventional designs rely on shallow depth ($L \leq 3$) and extremely wide hidden layers ($d_h \geq 3075$); and (3) **High-Dimensional Advantage:** Large latent dimensions ($d_z \geq 3075$) combined with small d_h improve performance, confirming that residual learning is more effective in wide latent spaces.

These results confirm that high-dimensional skip connections yield more expressive models with better performance–complexity trade-offs. As detailed in Appendix C, this robustness holds consistently across MNIST, ImageNet-32, and ImageNet-224, providing strong empirical support for the generality of the Hourglass principle.

Table 6: Pareto optimal model configurations for denoising task on ImageNet-224. An image is linearized to a vector of dimension $d_x = 3072$.

Architecture	**Params (M)**	d_z	d_h	L	**PSNR**
Conventional	37.767	3072	3075	1	23.210
	46.989	3072	4576	1	23.210
	62.448	3072	3546	2	24.704
	68.174	3072	4012	2	24.724
	75.553	3072	3075	3	24.939
	84.234	3072	3546	3	24.958
Hourglass	19.286	3075	16	4	23.592
	19.877	3075	16	10	23.767
	20.861	3075	16	20	23.961
	21.722	3075	115	4	24.065
	22.429	3075	115	5	24.172
	23.136	3075	115	6	24.272
	23.844	3075	115	7	24.347
	23.874	3075	270	3	24.359
	25.535	3075	270	4	24.446
	28.856	3075	270	6	24.627
	33.007	3075	765	3	24.643
	37.712	3075	765	4	24.780
	42.417	3075	765	5	24.859
	47.084	3075	1146	4	24.878
	47.121	3075	765	6	24.901
	54.339	3546	765	6	24.925
	61.480	4012	765	6	24.942
	66.041	3546	1560	4	24.959
	87.237	4012	1560	5	25.025

Table 7: Pareto optimal model configurations for super-resolution task on ImageNet-224. An image is linearized to a vector of dimension $d_x = 768$.

Architecture	**Params (M)**	d_z	d_h	L	**PSNR**
Conventional	30.689	3072	3075	1	26.785
	33.583	3072	3546	1	26.787
	49.582	3072	3075	2	27.836
	68.475	3072	3075	3	28.030
	77.156	3072	3546	3	28.030
	87.368	3072	3075	4	28.084
Hourglass	12.300	3075	16	5	27.301
	13.284	3075	48	5	27.475
	14.637	3075	115	4	27.516
	14.981	3075	86	6	27.521
	16.039	3075	86	8	27.593
	17.466	3075	115	8	27.696
	18.450	3075	270	4	27.723
	18.881	3075	115	10	27.763
	21.771	3075	270	6	27.860
	25.092	3075	270	8	27.920
	28.935	3546	270	8	27.931
	30.627	3075	765	4	27.965
	35.318	3546	765	4	27.980
	40.000	3075	1146	4	28.018
	46.169	3546	765	6	28.049
	57.871	3546	1560	4	28.069
	61.352	3075	2014	4	28.077
	69.372	3075	1560	6	28.086
	80.047	4012	2014	4	28.104

Appendix C. Additional Experiments and Results for Hourglass MLP

This appendix provides extended empirical evidence supporting the architectural advantages of the Hourglass MLP. While the main text (Sec. 3) focuses on high-resolution ImageNet-224, we here detail the comprehensive results for **MNIST** and **ImageNet-32**. These experiments cover generative classification, denoising, and super-resolution, alongside a systematic ablation of key architectural hyperparameters.

C.1. Experimental Setup

The experimental protocols for MNIST and ImageNet-32 follow the unified framework described in Sec. 2. Specifically, for MNIST, we introduce a *generative classification* task where the model is trained to map input digits to standard prototypical images (one fixed target per class). This evaluates the model's ability to learn high-level semantic mappings in the high-dimensional latent space d_z. Detailed hyperparameters for these smaller-scale experiments are summarized in Appendix A.

C.2. Main Results and Observations

We evaluate both architectures by characterizing their performance-parameter Pareto frontiers for each dataset and task combination. The Pareto frontier captures the trade-off between model complexity (number of parameters) and performance (measured by PSNR and SSIM). A model is Pareto-optimal if no other model achieves better performance with fewer parameters—these models represent the most efficient designs at their respective parameter budgets.

Our analysis reveals that Hourglass architectures consistently achieve superior Pareto frontiers compared to conventional designs across all tested tasks. As parameter budgets increase, the optimal Hourglass configurations favor deeper networks with wider latent dimensions but narrower bottleneck dimensions. Additionally, while Hourglass architectures inherently require dimensional expansion for optimal performance, we observe that conventional MLPs can also benefit from random input projections that preserve dimensionality.

C.2.1. GENERATIVE CLASSIFICATION TASK

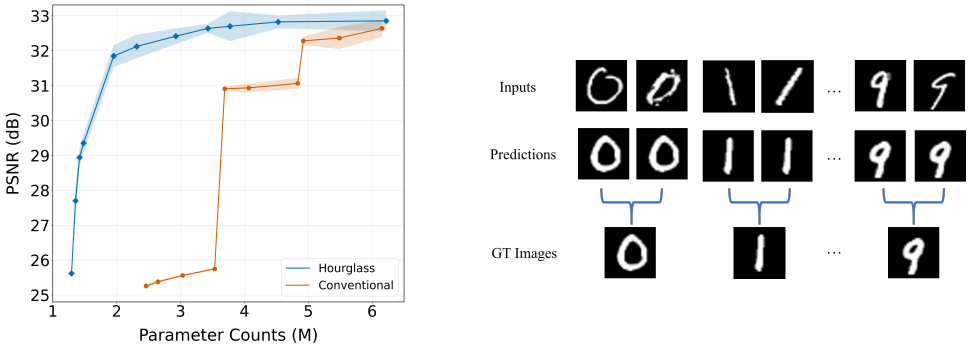
An MNIST generative classification task requires a model to take in an input digit image, generates a prototypical digit image, and then makes a classification based on the latter. Figure 6(b) shows qualitative examples from the Hourglass model. For model training, one image per digit class is chosen to serve as the ground truth digit image.

Figure 6(a) compares the Pareto frontiers of Hourglass and conventional MLPs on the MNIST generative classification task. As shown in Figure 6(a), the Hourglass architecture consistently achieves a better performance–complexity trade-off, reaching higher PSNR values across a wide range of parameter counts. In particular, when the required accuracy is low in the 26 dB range, the Hourglass architecture achieves superior performance with significantly fewer parameters.

C.2.2. GENERATIVE RESTORATION TASKS

We evaluate both architectures on two common generative restoration tasks: denoising and super-resolution. Figures 7 and 8 present the PSNR–parameter Pareto fronts for MNIST and ImageNet-32.

Across datasets and tasks, the proposed wide–narrow–wide (Hourglass) MLP consistently outperforms the conventional narrow–wide–narrow baseline. In denoising (Figure 7(b)), the Hourglass



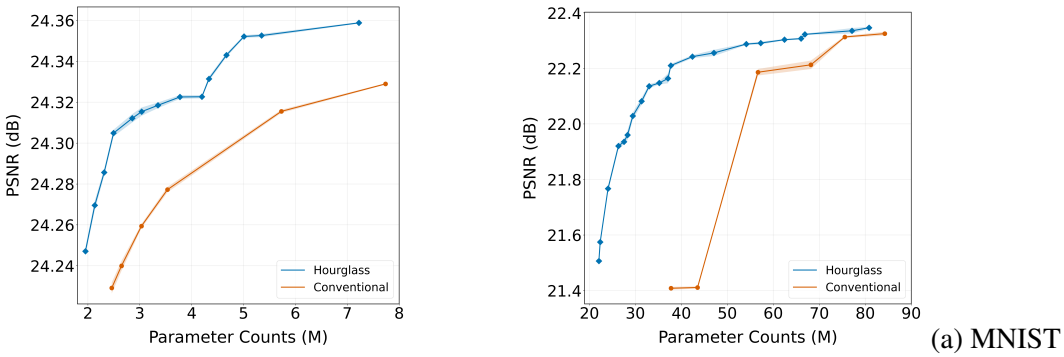
(a) (b)

Figure 6: **Generative Classification Task on MINST.** (a) Performance–complexity Pareto front. Fronts are searched with each configuration repeated 5 times. "Wide–narrow–wide" MLPs outperform conventional "narrow–wide–narrow" ones. (b) Samples predicted by our proposed Hourglass model.

model attains 22.31 dB PSNR with only 66M parameters, whereas the best conventional model requires 75M to reach the same score. On MNIST (Figure 7(a)), this advantage persists across the entire complexity range.

For super-resolution (Figure 8), the Hourglass design again dominates. On ImageNet-32, it achieves 24.00 dB with 69M parameters, outperforming the 87M-parameter conventional model. The gap is particularly pronounced in the mid-range budget regime. On MNIST, Hourglass MLPs similarly produce better reconstructions at every tested parameter count.

These results suggest that performing residual updates in high-dimensional latent space enhances restoration fidelity and parameter efficiency, especially under tight or mid-range model budgets.

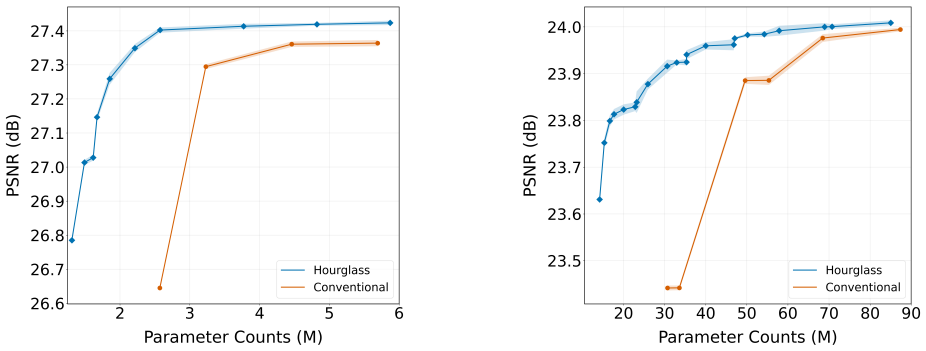


(b) ImageNet-32 (a) MNIST

Figure 7: **Generative Restoration Task - Denoising.** Performance-complexity Pareto fronts on MINST and ImageNet-32 are searched with each configuration repeated 5 times. Optimal configurations are shown in Table 8.

C.2.3. CONSISTENCY ACROSS DATASETS AND TASKS

As detailed in Tables 8 and 9, the architectural trends observed for ImageNet-32 and MNIST align consistently with the high-resolution ImageNet-224 results presented in the main text (Sec. 3.2). Specifically, we highlight the following consistent observations:



(a) MNIST (b) ImageNet-32

Figure 8: **Generative Restoration Task - Super-resolution.** Performance-complexity Pareto fronts on MINST and ImageNet-32 are searched with each configuration repeated 5 times. Optimal configurations are shown in Table 9.

- **Universal Efficiency:** Across all tested resolutions (from 28×28 to 224×224), the Hourglass MLP consistently yields superior Pareto frontiers, attaining equivalent or better fidelity with significantly fewer parameters than conventional baselines.
- **Optimal Scaling Robustness:** The optimal configurations across different datasets consistently favor increased network depth ($L \geq 4$) and expanded latent dimensions ($d_z \geq 3075$), while maintaining efficient narrow bottlenecks (d_h).
- **Empirical Validation of Design Intuition:** The persistent advantage of Hourglass models across various data complexities provides strong empirical support for our design intuition: performing residual refinement in higher-dimensional signal spaces enables more expressive and effective incremental updates.

These results confirm that the benefits of the Hourglass design principle are robust to changes in dataset scale and task type, establishing it as a versatile architectural improvement for MLP-based residual refinement.

Architecture	Params (M)	d_z	d_h	L	PSNR ($\mu \pm 5\sigma$ dB)
Conventional	37.77	3072	3075	1	21.408 ± 0.005
	43.52	3072	4012	1	21.411 ± 0.004
	56.66	3072	3075	2	22.186 ± 0.012
	68.17	3072	4012	2	22.213 ± 0.015
	75.55	3072	3075	3	22.313 ± 0.004
	84.23	3072	3546	3	22.325 ± 0.007
Hourglass	22.07	3546	8	5	21.506 ± 0.007
	22.35	3546	16	5	21.575 ± 0.012
	24.06	3546	64	5	21.767 ± 0.010
	26.33	3546	128	5	21.921 ± 0.010
	27.53	3546	270	3	21.936 ± 0.009
	28.30	3075	765	2	21.960 ± 0.017
	29.45	3546	270	4	22.029 ± 0.012
	31.36	3546	270	5	22.082 ± 0.012
	33.01	3075	765	3	22.136 ± 0.007
	35.19	3546	270	7	22.147 ± 0.005
	37.11	3546	270	8	22.164 ± 0.017
	37.71	3075	765	4	22.210 ± 0.006
	42.42	3075	765	5	22.242 ± 0.005
	47.08	3075	1146	4	22.256 ± 0.011
	54.13	3075	1146	5	22.288 ± 0.003
	57.27	3075	1560	4	22.291 ± 0.005
	62.42	3546	1146	5	22.303 ± 0.003
	66.04	3546	1560	4	22.307 ± 0.003
	66.86	3075	1560	5	22.323 ± 0.002
	77.10	3546	1560	5	22.335 ± 0.010
	80.82	3075	2014	5	22.346 ± 0.004

Table 8: Pareto optimal model configurations for denoising task on ImageNet-32. PSNR results are averaged over 5 independent runs. An image is linearized to a vector of dimension $d_x = 3072$.

Architecture	Params (M)	d_z	d_h	L	PSNR ($\mu \pm 5\sigma$ dB)
Conventional	30.69	3072	3075	1	23.442 ± 0.005
	33.58	3072	3546	1	23.442 ± 0.005
	49.58	3072	3075	2	23.885 ± 0.007
	55.37	3072	3546	2	23.886 ± 0.010
	68.48	3072	3075	3	23.976 ± 0.008
	87.37	3072	3075	4	23.994 ± 0.004
Hourglass	14.18	3546	16	5	23.631 ± 0.008
	15.32	3546	48	5	23.752 ± 0.010
	16.67	3546	86	5	23.799 ± 0.012
	17.70	3546	115	5	23.813 ± 0.011
	20.02	4012	115	5	23.823 ± 0.011
	22.83	4576	115	5	23.829 ± 0.009
	23.19	3546	270	5	23.839 ± 0.023
	25.92	3075	765	3	23.878 ± 0.012
	30.63	3075	765	4	23.916 ± 0.014
	32.95	3075	1146	3	23.923 ± 0.004
	35.32	3546	765	4	23.925 ± 0.007
	35.33	3075	765	5	23.941 ± 0.010
	40.00	3075	1146	4	23.960 ± 0.008
	46.81	3546	1560	3	23.962 ± 0.012
	47.05	3075	1146	5	23.975 ± 0.002
	50.18	3075	1560	4	23.983 ± 0.004
	54.25	3546	1146	5	23.984 ± 0.006
	57.87	3546	1560	4	23.994 ± 0.003
	68.93	3546	1560	5	24.000 ± 0.002
	70.75	3546	2014	4	24.001 ± 0.006
	85.03	3546	2014	5	24.009 ± 0.004

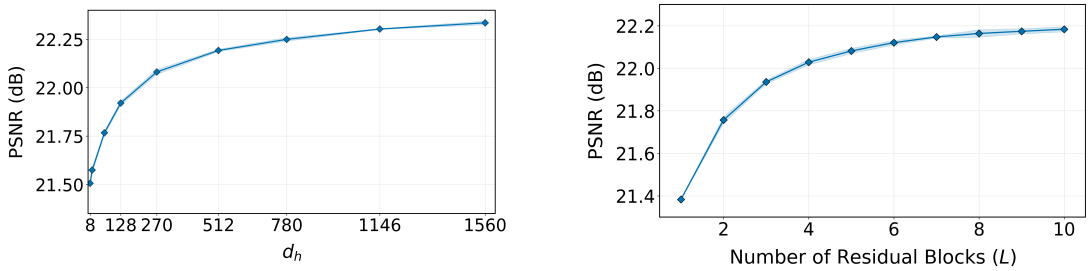
Table 9: Pareto optimal model configurations for super-resolution task on ImageNet-32. PSNR results are averaged over 5 independent runs. An image is linearized to a vector of dimension $d_x = 768$.

C.3. Ablation Studies on Hourglass MLP Design

To further explore the design trade-offs within the proposed wide–narrow–wide (Hourglass) MLP architecture, we conduct ablation studies focusing on two key hyperparameters: the bottleneck dimension d_h and the number of residual blocks L.

Effect of bottleneck width d_h: We fix the high-dimensional residual space to $d_z = 3546$ and the number of residual blocks to $L = 5$, and vary the bottleneck width d_h. As shown in Figure 9(a), increasing d_h improves PSNR, but the gains diminish beyond $d_h = 270$. This suggests that moderate bottlenecks are sufficient for high performance, enabling significant parameter savings.

Effect of residual depth L: We fix $d_z = 3546$, $d_h = 270$, and vary the number of residual blocks L. As shown in Figure 9(b), performance improves with deeper stacks, but quickly plateaus around $L = 5$, indicating that relatively shallow Hourglass MLPs are sufficient for strong results.



(a) Varying the bottleneck dimension d_h

(b) Varying the number of residual blocks L

Figure 9: Ablation study of optimal d_h and L for the Hourglass MLP architecture.

Appendix D. Detailed Experimental Setup for Hourglass ViT

D.1. Code, Library, and Hardware

We use the official codebase released by Nakamura et al. [13][1] to train the ViT models. Specifically, we use the standard Pytorch Image Models library (*timm*) [19], implemented in HuggingFace [20], to load the ViT backbones:

- ViT-Tiny: *vit_tiny_patch16_224*
- ViT-Small: *vit_small_patch16_224*
- ViT-Large: *vit_large_patch16_224*

We use the default model configuration options in *timm* to modify the model architectures, including d_z, d_h, L, and the MLP ratios reported in Table 2. Message Passing Interface (MPI) is used for parallel training across four RTX 3090 GPUs, with gradient accumulation to reduce memory overhead.

To calculate the training FLOPs reported in Table 2 for different model architectures, we use the official PyTorch Profiler[2]. For all ViT backbones, we attach the classification head corresponding to the CARS dataset, which has the largest number of classes among our datasets (196 classes). This setting accounts for the potentially larger additional training FLOPs introduced by W_{out} in the Hourglass architecture.

D.2. Datasets

Table 10 summarizes the dataset statistics for the five benchmarks used in our ViT training experiments. With 1000 training epochs, the total training throughput is already high, particularly for ImageNet-100, where the throughput reaches 117,000k. This suggests that the ViT models were trained sufficiently thoroughly. For ImageNet-100, we adopt the openly released dataset hosted on Hugging Face[3].

Table 10: Dataset statistics for the five image classification benchmarks. Numbers are reported in thousands (k), with training throughput being computed as the number of training samples multiplied by a total of 1000 training epochs.

Dataset	Training Samples (k)	Training Throughput (k)	Testing Samples (k)	Classes
CARS	8.2	8144	8.0	196
CIFAR-10	50.0	50000	10.0	10
CIFAR-100	50.0	50000	10.0	100
Flowers	6.1	6149	1.0	102
ImageNet-100	117.0	117000	13.0	100

1. https://github.com/ryoo-nakamura/OFDB
2. https://github.com/pytorch/pytorch/blob/main/torch/profiler/profiler.py
3. https://huggingface.co/datasets/ilee0022/ImageNet100

D.3. Training Hyperparameters

For both conventional and hourglass ViTs, we adopt training configurations similar to Nakamura et al. [13] across all model scales, applying identical hyperparameters to both conventional and hourglass architectures. A summary of the training hyperparameters is presented in Table 11.

Table 11: Training hyperparameters for ViT. Notably, for ViT-Large, the learning rate and batch size are halved to ensure convergence (reducing the learning rate from 4.5×10^{-2} to 2.25×10^{-2} and the batch size from 768 to 384).

Hyperparameter	**Value**
Epochs	1000
Learning Rate	4.5×10^{-2}
Batch Size	768
Optimizer	SGD
LR Scheduler	Cosine annealing with warmup
Warmup Epochs	10
Weight Decay	1.0×10^{-4}
Resolution	224×224
Label Smoothing	0.1
Drop Path	0.1
Rand Augment	(9, 0.5)
Mixup	0.8
Cutmix	1.0
Random Erasing	0.25

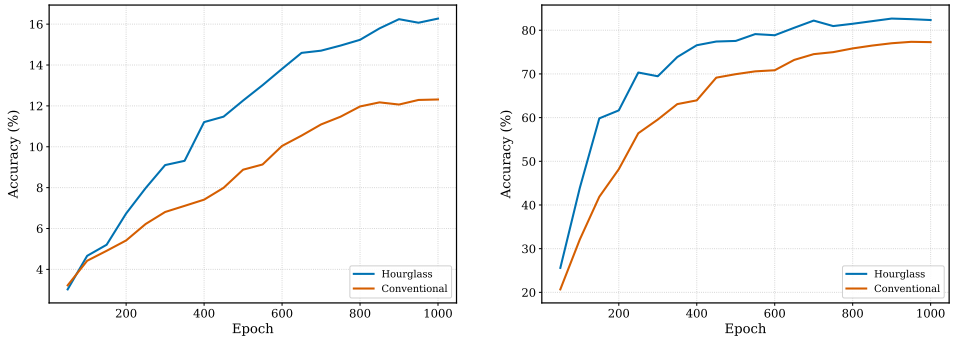
Appendix E. Additional Experiments and Results for Hourglass ViT

This section provides a detailed analysis of the training dynamics and performance consistency of the Hourglass ViT across various scales and benchmarks, supplementing the summary results in Section 3.

E.1. Training Dynamics: Hourglass vs. Conventional ViT

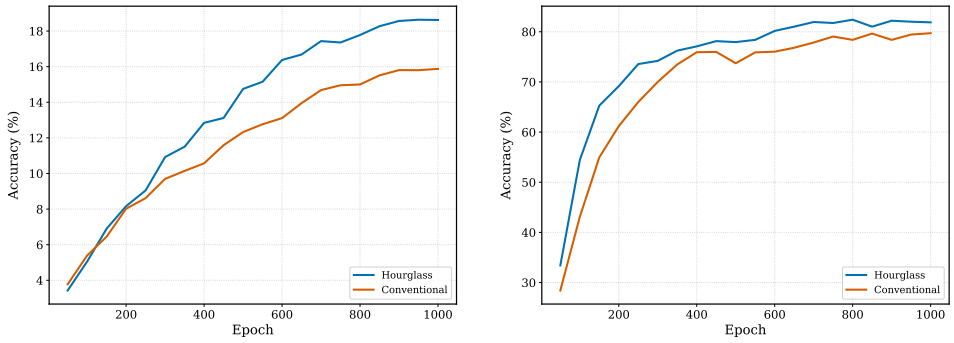
To further evaluate the optimization efficiency of our design, we visualize the training trajectories for both architectures. Figure 10, Figure 11 and Figure 12 present the top-1 accuracy curves on *Stanford Cars* and *Oxford Flowers-102* for the *Tiny*, *Small* and *Large* scale, respectively.

We observe that the Hourglass ViT not only attains higher final accuracy but also exhibits a more stable and efficient learning process during the early-to-mid training stages. This suggests that performing relational modeling (MHA) and incremental updates within the expanded residual space d_z provides a more favorable optimization landscape, especially for smaller model configurations.



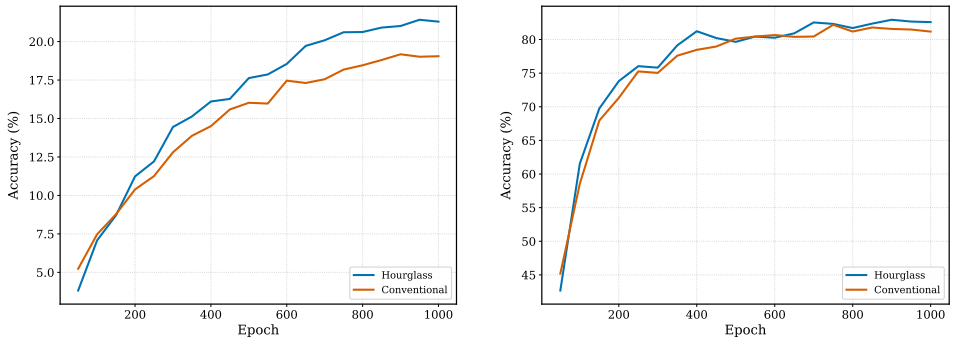
(a) Stanford Cars (b) Oxford Flowers-102

Figure 10: **Training curves comparison (Tiny scale).** Hourglass ViT demonstrates faster accuracy acquisition and superior final performance compared to the conventional baseline.



(a) Stanford Cars (b) Oxford Flowers-102

Figure 11: **Training curves comparison (Small scale).** Hourglass ViT demonstrates faster accuracy acquisition and superior final performance compared to the conventional baseline.



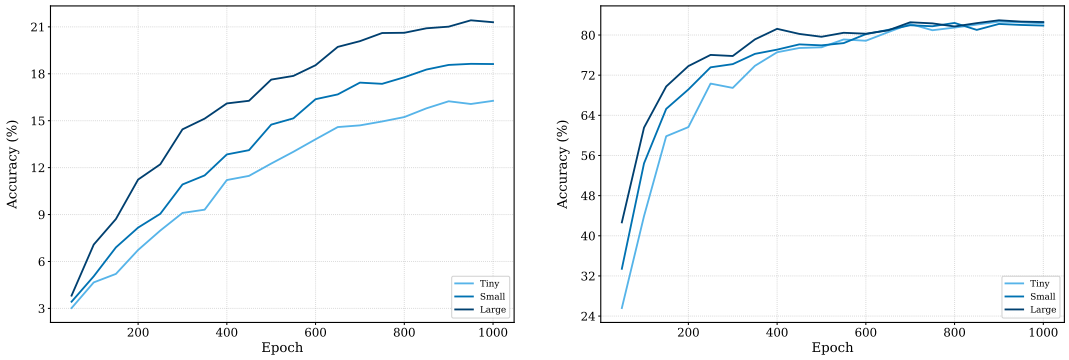
(a) Stanford Cars (b) Oxford Flowers-102

Figure 12: **Training curves comparison (Large scale).** Hourglass ViT demonstrates faster accuracy acquisition and superior final performance compared to the conventional baseline.

E.2. Convergence Analysis Across Model Scales

Figure 13 illustrates the training dynamics of the Hourglass architecture across *Tiny*, *Small*, and *Large* scales. Consistent with our observations in section 3, there is a clear correlation between model scale and convergence efficiency.

Specifically, on the Oxford Flowers dataset (Figure 13b), the *Hourglass-Large* variant reaches its performance plateau significantly earlier than its smaller counterparts. This accelerated convergence in the large-scale regime indicates that as the parameter budget increases, the Hourglass design's strategy of reallocating parameters toward a wider latent space and increased depth effectively facilitates rapid and stable incremental refinement.



(a) Hourglass ViT on Cars (b) Hourglass ViT on Flowers

Figure 13: **Hourglass ViT convergence across scales.** Larger configurations (Small and Large) exhibit a notably steeper convergence slope, reaching peak performance faster than the Tiny variant.