Investigating the Role of Fine-Tuning in Addressing the Gap Between Synthetic and Real Data in Generative Foundation Models

Leonhard Hennicke[1] Christian Medeiros Adriano[1]

Holger Giese[1] Lukas Schott[2] Jan Mathias Koehler[2]

Abstract

Generative foundation models like Stable Diffusion have shown potential for transfer learning in computer vision, particularly in training student models for downstream tasks using generated data. However, these student models often exhibit a significant drop in accuracy compared to models trained on real data. In this research, we investigate the causes of this drop, focusing on the role of different layers in the student model. Our findings reveal that the drop mainly stems from the model's final layers. Building upon our insights, we investigate the data-efficiency of fine-tuning a synthetically trained model with real data applied to only the last layers. Our results suggest an improved trade-off between the amount of real training data used for fine-tuning and the model's accuracy, contributing to the understanding of the gap between synthetic and real data and indicating potential solutions to mitigate the scarcity of labeled real data.

1 Introduction

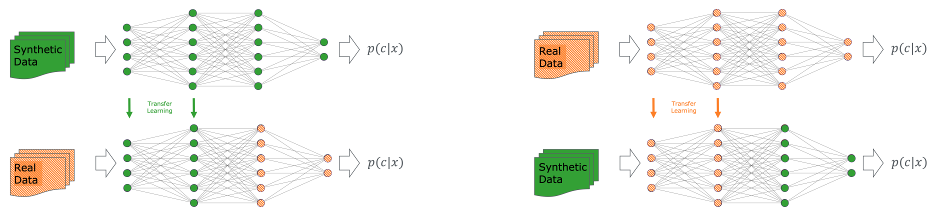
The rapid advancements in deep learning have significantly impacted computer vision tasks, yet they have also brought forth challenges such as the need for substantial amounts of labeled training data [3] and the computational resources required for deployment [23, 33, 17]. In response to these challenges, generative foundation models [4] like Stable Diffusion [24] have emerged as promising tools for transfer learning in computer vision, particularly in training student models for downstream tasks using generated data (data-free knowledge distillation [15]). However, a notable issue arises when these student models exhibit a significant drop in accuracy compared to models trained on real data [6] .

This research aims to delve into the causes of the accuracy drop in student models trained on generated data, with a specific focus on the role of different layers within the model. Our findings reveal that the drop in accuracy primarily originates from the model's final layers, shedding light on the potential for fine-tuning specific layers with real data to improve performance.

Building upon these insights, we investigate the data-efficiency of fine-tuning a synthetically trained model with real data applied to only the last layers. Our results suggest an improved trade-off between the amount of real training data used for fine-tuning and the model's accuracy, contributing to a deeper understanding of the gap between synthetic and real data. Furthermore, our findings indicate potential solutions to mitigate the scarcity of labeled real data, offering valuable perspectives on the principles and scalability of fine-tuning in modern machine learning.

[1]Hasso Plattner Institute, University of Potsdam, firstname.lastname@hpi.de

[2]Center for AI, Robert Bosch, firstname.lastname@de.bosch.com. Joint senior authors.



(a) Synthetic Data Transfer Learning. (b) Real Data Transfer Learning.

Figure 1: This figure illustrates our transfer learning setup for N = 2. N indicates the number of consecutive layers that are transferred from a model that was trained on the respective first dataset, starting from the first layer.

2 State of the Art in Data-Free Knowledge Distillation of Foundation Models

The current works demonstrate a significant gap in ImageNet-1k [6] top-1 accuracy when models are trained solely on synthetic data compared to additional fine-tuning on real data. With models trained exclusively on synthetic data achieving top-1 accuracies of at most 42.9%. In contrast, CLIP, trained on real data [22], achieves 68.6% zero-shot accuracy on a ViT-B/16 and 59.6% on a ResNet50 [9]. This accuracy gap is puzzling, given that synthetic images from models like StableDiffusion are, for humans, nearly indistinguishable from real ones.

Our paper is based on the Fake It Till You Make It (FITYMI) pipeline [27], training models on image data generated by StableDiffusion [24]. We adapt the FITYMI setup for our experiments to make our results directly comparable. We focus on using the ImageNet-100 dataset due to resource constraints, and even with this reduced dataset, naive training without augmentations leads to a low accuracy of 28.4%. The FITYMI paper covers ablations on ImageNet-100, including variations of prompts, augmentations, and the guidance scale, which significantly impact performance. Despite these optimizations, they only achieve a zero-shot accuracy of 42.9%, as a large portion of the accuracy gap remains. We extend the ablations by testing out the importance of fine-tuning the last layers with real data.

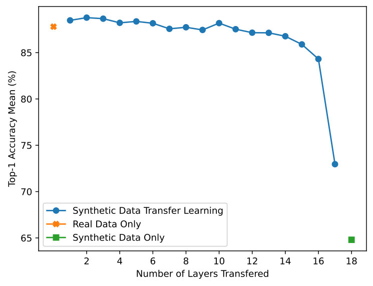
Similarly, StableRep [30] presents a pipeline learning representations solely based on synthetic images, achieving a zero-shot top-1 accuracy of 34.9% compared to 73.7% linear probing with a ViT-B/16 [7]. They use captions from datasets to generate image data with Stable Diffusion and train the model in an unsupervised manner. SynthCLIP [8] achieves an even lower zero-shot accuracy of 30.5%. Lastly, Azizi et al. [2] use Imagen [26] to generate synthetic training data for image classification, achieving a top-1 accuracy of 69.24% on ImageNet using a ResNet50 trained for 90 epochs, although their approach involves fine-tuning ImageNet data, making it not completely data-free.

3 Experiments

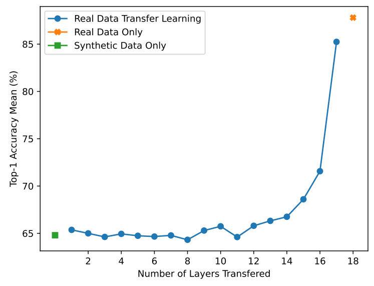
3.1 Experimental Setup

Formal definitions of Stable Defusion and Data-Free Knowledge Distillation are described in section A. Our experiments are inspired by the findings of the paper Fake It Till You Make It (FITYMI) [27]. To ensure comparability, we used the same setup as in FITYMI for all our experiments, including training a randomly initialized network for 100 epochs using specific optimization techniques and augmentations. We also provide results on the Oxford-IIIT Pet dataset [19] using TinyViT-5M [32], a vision transformer that incorporates convolutional layers. The synthetic data for Pets was generated by Stable Diffusion XL [20], using the same methods as in [21] for comparability. We report the mean and standard deviation on real ImageNet-100 validation data of the last 5 epochs for top-1 and top-5 accuracy. Further details and exact cropping parameters are provided in the supplemental material, section B.

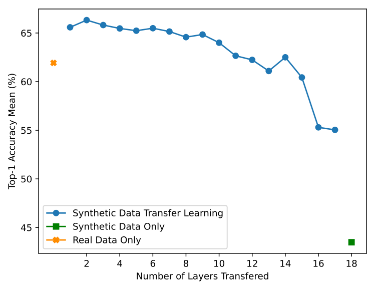
We conducted a series of experiments using transfer learning to identify which layers are responsible for the bottleneck in performance when training with merely synthetic data. We pre-trained a CNN



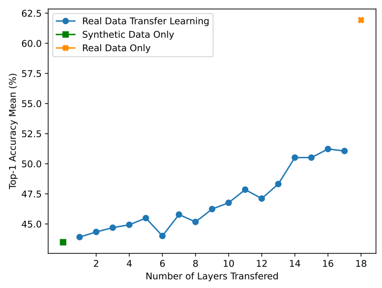
(a) Synthetic Data TL, ImageNet.



(b) Real Data TL, ImageNet.



(c) Synthetic Data TL, Pets.



(d) Real Data TL, Pets.

Figure 2: Results of the layer importance experiments using transfer learning (TL) from real and synthetic data for Pets and ImageNet. We also plot the results for our baseline models trained on synthetic and real data only, as these present the two extremes of this experiment setup and therefore provide an approximate lower and upper bound.

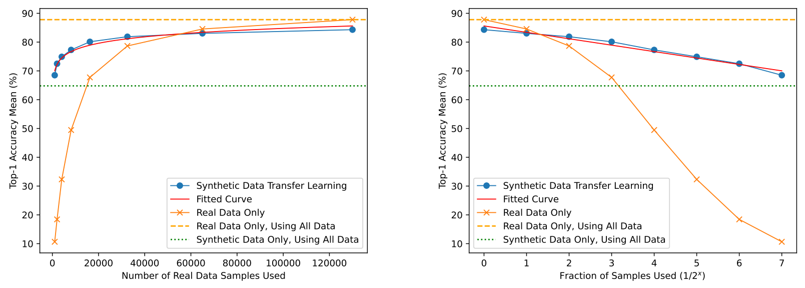
on either synthetic or real data, froze[1] the first N layers, and reinitialized the remaining layers for retraining on the respective other dataset. By gradually increasing N, we expected to observe changes in the accuracy of the resulting model, reflecting the quality of features at gradually higher levels of abstraction present in the pre-trained data. These experiments were performed on ImageNet-100 and Pets datasets. In the context of our experiments, layers in the ResNet-50 architecture for ImageNet refer to the bottleneck blocks; for Pets, layers are defined by each convolutional block, each TinyViT block and each downsampling block, as well as the initial patch embedding block, the final classification layer and the preceding normalization layer; amounting to 18 layers total. Running this setup for N = 17 results in a setup equivalent to linear probing, while N = 18 would result in evaluating a completely pre-trained model with all parameters frozen.

3.2 Layer Importance Experiments

The Gap between synthetic and real: The results of this experiment are presented in Figure 2. We directly see the gap between synthetic and real by comparing models solely trained on real and synthetic data (orange and green squares). Note, that our baselines on Pets are less accurate than on ImageNet-100. We suspect that this is due to the smaller number of samples (37 classes $*$ 100 samples) in the training data compared to ImageNet-100. To show the validity of our training pipeline, we also provide baselines for real and synthetic data using a TinyViT-5M pre-trained on ImageNet-1K, which is commonly done for this dataset [14, 18]. Here, we see accuracies for real data 93.1% and synthetic data 80.2% (not shown in figure), confirming the gap. Our baselines without pre-training achieve similar accuracies to other state-of-the-art models [29] when training from scratch.

Synthetic Data Transfer Learning: In our experiments, we transfer the first N layers from a model pre-trained on synthetic data with frozen parameters and train the remaining layers on real data (illustrated in Figure 1a for $N = 2$). We evaluate the performance on real data.

[1]By freezing we mean that no gradients are calculated for these layers and no updates are made to their parameters during training from that point on.



(a) Linear Scale (Read Right to Left) (b) Logarithmic Scale (Read Left to Right)

Figure 3: **Data Reduction Experiments.** Top-1 accuracy of reducing the amount of real training data with (blue) and without (orange) synthetic data transfer learning (SDTL). For transfer learning, the first 16 layers are trained on synthetic data and frozen. The last two layers are fine-tuned with a different number of real data samples. The fitted curves (red) are derived from the top-1 accuracy of SDTL on the respective scales via curve fitting, using least squares polynomial fitting on a 1st-degree polynomial ($f(x) = -2.23log(x) + 85.61$).

The accuracy does not gradually drop as more layers are replaced with those pre-trained on synthetic data, indicating that the quality of the representations learned from synthetic data varies with the level of abstraction. For ImageNet, the top-1 accuracy only decreases by 3.5 pp when pre-training layers 1 to 16 with synthetic data (Figure 2a), and even increases by 1.0pp with some in-between pre-trained layers. A larger decrease of 11.3pp in top-1 accuracy is visible when the 17th layer is replaced by one pre-trained on synthetic data (Figure 2a). A similar drop is observed for Pets in later layers (>15) in Figure 2c.

Real Data Transfer Learning: To further investigate our findings on ImageNet, we conducted a second series of experiments similar to the previous section. In this setup, the first N layers are transferred from a model pre-trained on real data with frozen parameters, while the remaining layers are reinitialized and trained on synthetic data (illustrated in Figure 1b for N = 2). The evaluation is always performed on real validation data. The results of this experiment for ImageNet are presented in Figure 2b. We found that the top-1 accuracy increases by at most 2.0pp from the baseline when adding the pre-trained layers 1 to 14, and even decreases for some in-between layers. However, when adding the pre-trained layers 15 to 17, we observed progressively larger increases in top-1 accuracy of 18.4pp in total. This effect is not equally visible for Pets, as shown in Figure 2d.

Data-Reduction Experiments: Based on our layer importance experiments, we suspect that training only the last layers on real data (with the earlier layers pre-trained on synthetic data and then frozen) would require less real data to achieve similar accuracy as a fully real data-trained baseline model. To test this, we pre-train the first 16 layers on synthetic data and then reinitialize and retrain the remaining layers on a randomly drawn, reduced number of real ImageNet-100 samples, for the same number of epochs. We expect the accuracy to remain within a similar range as the baseline model, even with reduced real training data. The results in Figure 3 show that even with an exponential reduction in real training data ($1/2^x$), the top-1 accuracy only decreases by 4.2pp after the $x = 3$rd iteration (using 1/8th of the entire real training data). The fitted curve suggests a logarithmic decrease in top-1 accuracy as real training data is reduced. Additionally, we observe that the drop in accuracy for a now randomly initialized model is steeper than for models with frozen layers pre-trained on synthetic data, indicating that synthetic pre-training can mitigate the accuracy reduction caused by a lack of real data.

4 Discussion

We investigated the accuracy gap between models trained on synthetic data vs. fine-tuning partly on real data. Our contributions are twofold: (1) we find that mostly the final synthetically trained layers are responsible for the drop in accuracy from synthetic to real, e.g., within a ResNet-50 the training of all but the last two layers can be done with only synthetic data, resulting in a minor accuracy drop; (2) we show that one could pre-train and then freeze all but the last two layers merely with synthetic data and fine-tune the remaining layers with a fraction of real data, e.g., using 1/8 of real data on a pre-trained model drops performance by 7pp compared to 20pp when omitting pre-training on synthesized data

Acknowledgments and Disclosure of Funding

This work was supported in part by Bosch Center for Artificial Intelligence and the Hasso-Plattner-Institute. We thank Claudia Blaiotta, Bastian Bischoff, Niclas Popp, Anna Khoreva, and Jan H. Metzen for their discussions and feedback on the manuscript. The Bosch Group is carbon neutral. Administration, manufacturing and research activities do no longer leave a carbon footprint. This also includes GPU clusters on which most experiments have been performed.

References

[1] AI, M.: Pytorch examples. http://github.com/pytorch/examples (2013)

[2] Azizi, S., Kornblith, S., Saharia, C., Norouzi, M., Fleet, D.J.: Synthetic data from diffusion models improves imagenet classification. arXiv:2304.08466 (2023)

[3] Bhardwaj, K., Suda, N., Marculescu, R.: Edgeal: A vision for deep learning in the iot era. IEEE Design & Test **38**(4), 37–43 (2019)

[4] Bommasani, R., Hudson, D.A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M.S., Bohg, J., Bosselut, A., Brunskill, E., et al.: On the opportunities and risks of foundation models. arXiv:2108.07258 (2021)

[5] Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging Properties in Self-Supervised Vision Transformers. In: ICCV Intl. Conf. on Computer Vision (2021)

[6] Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet a large-scale hierarchical image database. In: CVPR Conf. on computer vision and pattern recognition. pp. 248–255 (2009)

[7] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv:2010.11929 (2020)

[8] Hammoud, H.A.A.K., Itani, H., Pizzati, F., Torr, P., Bibi, A., Ghanem, B.: Synthclip: Are we ready for a fully synthetic clip training? arXiv:2402.01832 (2024)

[9] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR Conf. on Computer Vision and Pattern Recognition. pp. 770–778 (2016)

[10] Hendrycks, D., Mu, N., Cubuk, E.D., Zoph, B., Gilmer, J., Lakshminarayanan, B.: AugMix: A Simple Data Processing Method to Improve Robustness and Uncertainty. ICLR Intl. Conf. on Learning Representations (2020)

[11] Hendrycks, D., Zou, A., Mazeika, M., Tang, L., Li, B., Song, D., Steinhardt, J.: PixMix: Dreamlike Pictures Comprehensively Improve Safety Measures. In: CVPR Conf. on Computer Vision and Pattern Recognition (2022)

[12] Hinton, G., Vinyals, O., Dean, J., et al.: Distilling the knowledge in a neural network. arXiv:1503.02531 (2015)

[13] Ho, J., Salimans, T.: Classifier-free diffusion guidance. arXiv:2207.12598 (2022)

[14] Hong, J., Huang, K., Liang, H.N., Wang, X., Zhang, R.: Fine-grained image classification with object-part model. In: Advances in Brain Inspired Cognitive Systems: 10th International Conference, BICS 2019, Guangzhou, China, July 13–14, 2019, Proceedings 10. pp. 233–243. Springer (2020)

[15] Lopes, R.G., Fenu, S., Starner, T.: Data-free knowledge distillation for deep neural networks. arXiv:1710.07535 (2017)

[16] Miller, G.A.: Wordnet: a lexical database for english. Communications of the ACM **38**(11), 39–41 (1995)

[17] Miotto, R., Wang, F., Wang, S., Jiang, X., Dudley, J.T.: Deep learning for healthcare: review, opportunities and challenges. Briefings in bioinformatics **19**(6), 1236–1246 (2018)

[18] Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., et al.: Dinov2: Learning robust visual features without supervision. arXiv preprint arXiv:2304.07193 (2023)

[19] Parkhi, O.M., Vedaldi, A., Zisserman, A., Jawahar, C.: Cats and dogs. In: 2012 IEEE conference on computer vision and pattern recognition. pp. 3498–3505. IEEE (2012)

[20] Podell, D., English, Z., Lacey, K., Blattmann, A., Dockhorn, T., Müller, J., Penna, J., Rombach, R.: Sdxl: Improving latent diffusion models for high-resolution image synthesis. arXiv preprint arXiv:2307.01952 (2023)

[21] Popp, N., Metzen, J.H., Hein, M.: Zero-shot distillation for image encoders: How to make effective use of synthetic data (2024)

[22] Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: ICML Intl. Conf. on Machine Learning. pp. 8748–8763 (2021)

[23] Ravi, D., Wong, C., Lo, B., Yang, G.Z.: A deep learning approach to on-node sensor data analytics for mobile or wearable devices. IEEE Journal of Biomedical and Health Informatics **21**(1), 56–64 (2016)

[24] Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: CVPR Conf. on Computer Vision and Pattern Recognition. pp. 10684–10695 (2022)

[25] Ruder, S.: An overview of gradient descent optimization algorithms. arXiv:1609.04747 (2016)

[26] Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E., Ghasemipour, S.K.S., Ayan, B.K., Mahdavi, S.S., Lopes, R.G., et al.: Photorealistic text-to-image diffusion models with deep language understanding. arXiv:2205.11487 (2022)

[27] Sarıyıldız, M.B., Alahari, K., Larlus, D., Kalantidis, Y.: Fake it till you make it: Learning transferable representations from synthetic imagenet clones. In: CVPR Conf. on Computer Vision and Pattern Recognition. pp. 8011–8021 (2023)

[28] Sariyildiz, M.B., Kalantidis, Y., Alahari, K., Larlus, D.: No reason for no supervision: Improved generalization in supervised models. arXiv preprint arXiv:2206.15369 (2022)

[29] Schuler, J.P.S., Romani, S., Abdel-Nasser, M., Rashwan, H., Puig, D.: Grouped pointwise convolutions reduce parameters in convolutional neural networks. In: Mendel. vol. 28, pp. 23–31 (2022)

[30] Tian, Y., Fan, L., Isola, P., Chang, H., Krishnan, D.: Stablerep: Synthetic images from text-to-image models make strong visual representation learners. Advances in Neural Information Processing Systems **36** (2024)

[31] Tian, Y., Krishnan, D., Isola, P.: Contrastive multiview coding. In: ECCV European Computer Vision Conf. pp. 776–794. Springer (2020)

[32] Wu, K., Zhang, J., Peng, H., Liu, M., Xiao, B., Fu, J., Yuan, L.: Tinyvit: Fast pretraining distillation for small vision transformers. In: European conference on computer vision. pp. 68–85. Springer (2022)

[33] Zhang, T., Gao, C., Ma, L., Lyu, M., Kim, M.: An empirical study of common challenges in developing deep learning applications. In: ISSRE Intl. Symp. on Software Reliability Engineering. pp. 104–115 (2019)

Supplementary Material

A Preliminaries

Definition 1 - Stable Diffusion a.k.a. latent diffusion model (*LDM*) is derived from a type of generative model [24] that employs equally weighted denoising autoencoders to predict the denoised version of input data, with an objective function minimizing the difference between the added and the predicted noise.

$$\mathcal{L}_{LDM} = \mathbb{E}_{\mathcal{E}(x),y,\epsilon\sim\mathcal{N}(0,1),t}\left[||\epsilon - \epsilon_\theta(z_t, t, \tau_\theta(y))||_2^2\right] \tag{1}$$

where τ_θ is the specialized encoder for the additional value guiding the generation process, which is trained jointly with ϵ_θ and y is the additional value. By shifting computations to a perceptually analogous but lower-dimensional domain through an auto-encoder, Stable Diffusion reduces the computational complexity because it can focus on the semantically significant aspects of data.

Definition 2 - Data-Free Knowledge Distillation precludes the access to the original training data for transferring knowledge from a teacher model to a student [15]. Instead, one leverages either the architecture or the learned parameters of the teacher model to distill knowledge to the student. This process often involves methods such as feature matching, activation mimicking, or utilizing generative models to generate synthetic data for distillation (our approach).

Definition 3 - Knowledge Distillation consists of transferring knowledge from a larger teacher model to a smaller student model while maintaining performance and reducing computational costs [12]. Initially, the teacher model is trained on a large dataset to achieve high accuracy. Then, the student model learns to mimic the teacher's behavior using both the original dataset and the teacher's soft targets, which contain probabilistic distribution insights. A Kullback-Leibler divergence loss function guides the student model to align its output distribution with that of the teacher model. equation and line below can be removed to save space

$$KL(\hat{y}, y) = y \cdot \log \frac{y}{\hat{y}} \tag{2}$$

where $\hat{y}$ and y are the predicted and true labels respectively.

Definition 4 - Foundation Models are deep neural networks (*DNN*) models with very large numbers of trainable parameters, which, thus, require orders of magnitude more data than traditional *DNN* models. Besides their prediction power, foundation models [4] are more adaptable to downstream tasks.

B Experimental Setup

The premise of our experiments is inspired by the findings of the paper Fake It Till You Make It (FITYMI) [27]. To make our results comparable with these findings, we used the same setup as in FITYMI in all of our experiments, i.e., training a randomly initialized network for 100 epochs using SGD [25] with a momentum of 0.9 and DINO augmentations with one global and eight local crops, unless stated otherwise. We linearly increased the learning rate up to 0.1 for the first 10 epochs and then decayed it on a cosine schedule. In addition, temperature scaling with L2-normalized weights was used on the final classification layer. Our training pipeline is based on the implementation of [28], which is publicly available and adapted for our classification task. For further details like exact cropping parameters are provided in the supplemental material. For most experiments, we use ImageNet-100 [31], a subset of ImageNet-1K [6]. This is the same subset of classes that is used in FITYMI. We use either the original data from ImageNet-100 or a synthetic version of this dataset with 1300 images generated per class. The synthetic images were generated using Stable Diffusion 1.4 unless stated otherwise. This was done using 50 diffusion steps and a guidance scale [13] of 2.0. The images were generated sized 512 x 384 and using the WordNet [16] based definition prompts from FITYMI (i.e., "<class name>, <WordNet definition of class name>"). Generating a full ImageNet-100 dataset with this setup takes over 73 single-GPU hours.

Table 1: **Augmentation Experiments.** The models were trained using different augmentation techniques, as specified in the *Experiment* column. To accurately portray the impact of the different augmentations, the baseline was trained using the FITYMI setup, as in our other experiments, but without the DINO augmentations. Hence, in this table, only the runs labeled "DINO" are equivalent to the full FITYMI setup (our baseline everywhere else).

Experiment	Top-1 Acc. (%)	Top-5 Acc. (%)	Train Loss
Real Data Only (R)	70.1 ±0.1	88.2 ±0.2	0.007 ±0.001
+ AutoAugment	79.7 ±0.1	94.6 ±0.1	0.033 ±0.001
+ PyTorch Example [1]	86.2 ±0.1	96.5 ±0.1	0.308 ±0.003
+ AugMix [10]	86.3 ±0.1	97.2 ±0.1	0.318 ±0.002
+ PixMix [11]	86.5 ±0.1	97.2 ±0.1	0.848 ±0.007
+ DINO [5]	87.8 ±0.1	97.1 ±0.1	0.742 ±0.002
Synth. Data Only (S)	46.9 ±0.2	71.7 ±0.2	0.006 ±0.001
+ AutoAugment	56.2 ±0.2	81.4 ±0.1	0.026 ±0.100
+ PyTorch Example [1]	59.5 ±0.1	83.9 ±0.2	0.261 ±0.002
+ AugMix [10]	60.3 ±0.2	84.5 ±0.2	0.281 ±0.003
+ PixMix [11]	61.8 ±0.2	84.7 ±0.2	0.680 ±0.006
+ DINO [5]	64.8 ±0.1	87.6 ±0.1	0.696 ±0.003

C Additional Details Regarding Model Training

Exact cosine schedule used for learning rate decay, given as a formula to calculate the learning rate for each each consecutive batch while the learning rate is decayed using this schedule, taken from [28]:

$$LR(i) = 0.000001 + 0.5 * (LR_{base} - 0.000001) * (1 + cos(\pi * i/i_{max})) \quad (3)$$

where i_{max} is the number of iterations the learning rate is decayed, i.e., the number of batches per epoch times the number of epochs the schedule is applied to, i is the current iteration and LR_{base} is the base learning rate of 0.1 multiplied by the batch size divided by 256. We use a batch size of 64 and 32 for ResNet-50 and TinyViT respectively.

For multicropping, the relative crop sizes are randomly sampled from the following intervals: $[0.25, 1.0]$ for global crops and $[0.05, 0.25]$ for local crops.

D Data Augmentations Experiments

As shown in FITYMI [27], the type of augmentations can make a difference for models trained on synthetic data, more so than when training on real data. Based on this, we suspected that the gap could be bridged through data augmentation. We test several sophisticated augmentation pipelines, both for models trained on real and on synthetic data. However, the accuracy gap still remains and we could also not replicate the effect of these augmentations w.r.t. leading to a larger improvement when training on synthetic data (as shown in FITYMI) compared to our baseline using no data augmentations. As shown in Table 1, we still observe this effect when comparing the specific augmentations used in FITYMI, i.e., the ones from the PyTorch example [1] and the ones from DINO [5]. However, the effect is less pronounced in our results than in the ones shown in FITYMI. The difference is likely due to a higher guidance scale [13] being set for Stable Diffusion in this specific experiment in FITYMI. Since higher guidance scales lead to less sampling diversity, their data contains less natural variations that could otherwise act similarly to data augmentations.

E Supplemental Tables for Sections 3.2 and 3.2

Table 2: **Synthetic Data Transfer Learning on ImageNet-100.** This table provides the concrete results for Figure 2a. Using the first N layers of the model pre-trained on synthetic ImageNet-100 data with frozen weights, the remaining layers were initialized randomly and trained on real ImageNet-100 data. We report the mean and standard deviation on real ImageNet-100 validation data of the last 5 epochs for top-1 and top-5 accuracy.

Frozen Layers	Top-1 Acc. (%)	Top-5 Acc. (%)	Train Loss
None - Synth. Data Only (S)	64.8 ±0.1	87.6 ±0.1	0.696 ±0.003
None - Real Data Only (R)	87.8 ±0.1	97.1 ±0.1	0.742 ±0.002
+ L. 0-1 from (S)	88.5 ±0.2	97.4 ±0.1	0.713 ±0.003
+ L. 0-2 from (S)	88.8 ±0.1	98.0 ±0.1	0.747 ±0.003
+ L. 0-3 from (S)	88.7 ±0.1	97.4 ±0.1	0.753 ±0.003
+ L. 0-4 from (S)	88.2 ±0.1	97.5 ±0.1	0.752 ±0.002
+ L. 0-5 from (S)	88.4 ±0.2	97.7 ±0.1	0.762 ±0.004
+ L. 0-6 from (S)	88.2 ±0.1	97.6 ±0.1	0.762 ±0.004
+ L. 0-7 from (S)	87.6 ±0.1	97.5 ±0.1	0.763 ±0.003
+ L. 0-8 from (S)	87.7 ±0.1	97.8 ±0.1	0.769 ±0.004
+ L. 0-9 from (S)	87.4 ±0.1	97.3 ±0.1	0.786 ±0.001
+ L. 0-10 from (S)	88.2 ±0.1	97.4 ±0.1	0.787 ±0.001
+ L. 0-11 from (S)	87.5 ±0.1	97.6 ±0.1	0.797 ±0.003
+ L. 0-12 from (S)	87.2 ±0.1	97.0 ±0.1	0.808 ±0.003
+ L. 0-13 from (S)	87.1 ±0.2	97.1 ±0.1	0.824 ±0.003
+ L. 0-14 from (S)	86.8 ±0.1	96.7 ±0.1	0.822 ±0.002
+ L. 0-15 from (S)	85.9 ±0.2	96.8 ±0.1	1.010 ±0.002
+ L. 0-16 from (S)	84.3 ±0.2	96.0 ±0.1	1.277 ±0.002
+ L. 0-17 from (S)	73.0 ±0.2	92.1 ±0.1	2.104 ±0.004

Table 3: **Real Data Transfer Learning on ImageNet-100.** This table provides the concrete results for Figure 2b. Using the first N layers of the model pre-trained on real ImageNet-100 data with frozen weights, the remaining layers were initialized randomly and trained on synthetic ImageNet-100 data. We report the mean and standard deviation on real ImageNet-100 validation data of the last 5 epochs for top-1 and top-5 accuracy.

Frozen Layers	Top-1 Acc. (%)	Top-5 Acc. (%)	Train Loss
None - Real Data Only (R)	87.8 ±0.1	97.1 ±0.1	0.742 ±0.002
None - Synth. Data Only (S)	64.8 ±0.1	87.6 ±0.1	0.696 ±0.003
+ L. 0-1 from (R)	65.4 ±0.3	87.4 ±0.2	0.713 ±0.003
+ L. 0-2 from (R)	65.0 ±0.1	87.4 ±0.1	0.723 ±0.003
+ L. 0-3 from (R)	64.6 ±0.1	87.2 ±0.1	0.724 ±0.002
+ L. 0-4 from (R)	64.9 ±0.2	87.8 ±0.2	0.721 ±0.004
+ L. 0-5 from (R)	64.7 ±0.2	86.4 ±0.1	0.727 ±0.004
+ L. 0-6 from (R)	64.7 ±0.2	86.6 ±0.2	0.732 ±0.002
+ L. 0-7 from (R)	64.8 ±0.1	87.6 ±0.2	0.738 ±0.003
+ L. 0-8 from (R)	64.3 ±0.2	86.6 ±0.2	0.740 ±0.003
+ L. 0-9 from (R)	65.3 ±0.1	86.7 ±0.2	0.746 ±0.004
+ L. 0-10 from (R)	65.7 ±0.2	88.3 ±0.2	0.748 ±0.002
+ L. 0-11 from (R)	64.6 ±0.3	87.0 ±0.2	0.698 ±0.004
+ L. 0-12 from (R)	65.8 ±0.4	87.1 ±0.1	0.703 ±0.002
+ L. 0-13 from (R)	66.3 ±0.4	87.6 ±0.2	0.712 ±0.001
+ L. 0-14 from (R)	66.8 ±0.4	87.7 ±0.1	0.727 ±0.002
+ L. 0-15 from (R)	68.6 ±0.3	89.1 ±0.2	0.889 ±0.002
+ L. 0-16 from (R)	71.6 ±0.2	90.5 ±0.2	1.069 ±0.002
+ L. 0-17 from (R)	85.2 ±0.3	95.6 ±0.2	1.925 ±0.003

Table 4: **Synthetic Data Transfer Learning on Oxford-IIIT Pet.** This table provides the concrete results for Figure 2c. Using the first N layers of the model pre-trained on synthetic Oxford-IIIT Pet data with frozen weights, the remaining layers were initialized randomly and trained on real Oxford-IIIT Pet data. We report the mean and standard deviation on real Oxford-IIIT Pet validation data of the last 5 epochs for top-1 and top-5 accuracy.

Frozen Layers	Top-1 Acc. (%)	Top-5 Acc. (%)	Train Loss
None - Synth. Data Only (S)	43.5 ±0.5	76.2 ±0.4	0.006 ±0.006
None - Real Data Only (R)	61.9 ±0.1	89.1 ±0.1	0.009 ±0.009
+ L. 0-1 from (S)	65.6 ±0.1	90.6 ±0.1	0.006 ±0.006
+ L. 0-2 from (S)	66.3 ±0.1	90.9 ±0.1	0.005 ±0.005
+ L. 0-3 from (S)	65.8 ±0.2	90.8 ±0.1	0.005 ±0.005
+ L. 0-4 from (S)	65.5 ±0.1	90.8 ±0.1	0.009 ±0.009
+ L. 0-5 from (S)	65.2 ±0.1	90.7 ±0.1	0.006 ±0.006
+ L. 0-6 from (S)	65.5 ±0.2	90.0 ±0.1	0.004 ±0.004
+ L. 0-7 from (S)	65.2 ±0.1	90.6 ±0.1	0.003 ±0.003
+ L. 0-8 from (S)	64.6 ±0.1	89.6 ±0.1	0.004 ±0.004
+ L. 0-9 from (S)	64.8 ±0.1	90.0 ±0.1	0.004 ±0.004
+ L. 0-10 from (S)	64.0 ±0.0	89.6 ±0.1	0.003 ±0.003
+ L. 0-11 from (S)	62.7 ±0.1	89.8 ±0.1	0.002 ±0.002
+ L. 0-12 from (S)	62.2 ±0.1	88.9 ±0.1	0.003 ±0.003
+ L. 0-13 from (S)	61.1 ±0.1	88.4 ±0.1	0.004 ±0.004
+ L. 0-14 from (S)	62.5 ±0.1	89.4 ±0.1	0.005 ±0.005
+ L. 0-15 from (S)	60.4 ±0.2	88.1 ±0.1	0.005 ±0.005
+ L. 0-16 from (S)	55.3 ±0.2	84.7 ±0.1	0.008 ±0.008
+ L. 0-17 from (S)	55.0 ±0.1	84.8 ±0.1	0.008 ±0.008

Table 5: **Real Data Transfer Learning on Oxford-IIIT Pet.** This table provides the concrete results for Figure 2d. Using the first N layers of the model pre-trained on real Oxford-IIIT Pet data with frozen weights, the remaining layers were initialized randomly and trained on synthetic IOxford-IIIT Pet data. We report the mean and standard deviation on real Oxford-IIIT Pet validation data of the last 5 epochs for top-1 and top-5 accuracy.

Frozen Layers	Top-1 Acc. (%)	Top-5 Acc. (%)	Train Loss
None - Real Data Only (R)	61.9 ±0.1	89.1 ±0.1	0.009 ±0.009
None - Synth. Data Only (S)	43.5 ±0.5	76.2 ±0.4	0.006 ±0.006
+ L. 0-1 from (S)	43.9 ±0.5	77.2 ±0.3	0.004 ±0.004
+ L. 0-2 from (S)	44.3 ±0.4	77.0 ±0.3	0.004 ±0.004
+ L. 0-3 from (S)	44.7 ±0.4	77.0 ±0.2	0.005 ±0.005
+ L. 0-4 from (S)	44.9 ±0.4	78.4 ±0.4	0.005 ±0.005
+ L. 0-5 from (S)	45.5 ±0.4	78.2 ±0.4	0.005 ±0.005
+ L. 0-6 from (S)	44.0 ±0.5	78.0 ±0.4	0.004 ±0.004
+ L. 0-7 from (S)	45.8 ±0.4	80.2 ±0.2	0.005 ±0.005
+ L. 0-8 from (S)	45.2 ±0.4	79.1 ±0.3	0.007 ±0.007
+ L. 0-9 from (S)	46.2 ±0.4	80.7 ±0.2	0.007 ±0.007
+ L. 0-10 from (S)	46.8 ±0.4	80.0 ±0.4	0.007 ±0.007
+ L. 0-11 from (S)	47.9 ±0.2	80.9 ±0.3	0.006 ±0.006
+ L. 0-12 from (S)	47.1 ±0.4	80.6 ±0.3	0.007 ±0.007
+ L. 0-13 from (S)	48.3 ±0.4	81.1 ±0.3	0.006 ±0.006
+ L. 0-14 from (S)	50.5 ±0.4	83.6 ±0.2	0.007 ±0.007
+ L. 0-15 from (S)	50.5 ±0.3	83.0 ±0.2	0.005 ±0.005
+ L. 0-16 from (S)	51.2 ±0.1	84.4 ±0.2	0.004 ±0.004
+ L. 0-17 from (S)	51.1 ±0.2	84.4 ±0.1	0.004 ±0.004

Table 6: **Data Reduction Experiments.** This table provides the concrete data points for Figure 4. The models in Table 6a (except for the baselines) were trained with the first 16 layers of the model pre-trained on synthetic ImageNet-100 data with frozen weights and the remaining layers being initialized randomly and trained on real ImageNet-100 data. The real training data used to train the last two layers was reduced to the fraction specified in the table using random sampling. In Table 6b we show the results of the same data reduction applied to models that are trained on real data only. We report the mean and standard deviation on real ImageNet-100 validation data of the last 5 epochs for top-1 and top-5 accuracy.

(a) Synthetic Data Transfer Learning

Experiment	Top-1 Acc. (%)	Top-5 Acc. (%)	Train Loss
Real Data Only (R)	87.8 ±0.1	97.1 ±0.1	0.742 ±0.002
Synth. Data Only (S)	64.8 ±0.1	87.6 ±0.1	0.696 ±0.003
+ 1/1 of real data used	84.3 ±0.2	96.0 ±0.1	1.277 ±0.002
+ 1/2 of real data used	83.0 ±0.1	96.2 ±0.1	1.313 ±0.004
+ 1/4 of real data used	81.9 ±0.2	95.2 ±0.1	1.398 ±0.005
+ 1/8 of real data used	80.1 ±0.2	94.7 ±0.1	1.477 ±0.006
+ 1/16 of real data used	77.3 ±0.3	93.8 ±0.1	1.533 ±0.007
+ 1/32 of real data used	74.9 ±0.3	93.0 ±0.1	1.558 ±0.006
+ 1/64 of real data used	72.5 ±0.2	91.6 ±0.1	1.616 ±0.021
+ 1/128 of real data used	68.5 ±0.2	90.8 ±0.2	1.516 ±0.039

(b) Real Data Only

Experiment	Top-1 Acc. (%)	Top-5 Acc. (%)	Train Loss
1/1 of real data used	87.8 ±0.1	97.1 ±0.1	0.742 ±0.002
1/2 of real data used	84.6 ±0.1	99.0 ±0.1	0.822 ±0.002
1/4 of real data used	78.7 ±0.1	94.1 ±0.1	1.124 ±0.002
1/8 of real data used	67.7 ±0.1	89.0 ±0.2	1.529 ±0.002
1/16 of real data used	49.5 ±0.2	78.2 ±0.2	2.179 ±0.004
1/32 of real data used	32.3 ±0.1	61.9 ±0.1	2.808 ±0.005
1/64 of real data used	18.4 ±0.1	43.8 ±0.2	3.254 ±0.009
1/128 of real data used	10.7 ±0.1	28.6 ±0.2	3.454 ±0.005