# DeepWeightFlow: Re-Basined Flow Matching for Generating Neural Network Weights

Saumya Gupta[1], Scott Biggs[2], Moritz Laber[3], Zohair Shafi[2], Robin Walters[2], Ayan Paul[1]
[1]Institute for Experiential AI, [2]Khoury College of Computer Science, [3]Network Science Institute, Northeastern University
{gupta.saumy, biggs.s, laber.m, shafi.z, r.walters, a.paul}@northeastern.edu

ICLR International Conference On Learning Representations

## Motivation

### Weight Space Generative Models

Neural network weights are an emerging data modality for many learning tasks, including generative modeling. Generating neural network weights is challenging due to the weight space symmetries. We combine canonicalization and flow matching to obtain an efficient generative model, *DeepWeightFlow,* for generating complete, high-quality weights for neural networks.

## Goals

1. We use canonicalization to address permutation symmetries.
2. Validate **complete** weight generation **without** fine tuning against SOTA methods.
3. Improve the training and inference speeds of weight generation using flow matching.
4. Demonstrate the viability of generating up to $\mathcal{O}(100M)$ parameters using dual PCA.
5. Show weight generation works for image and well as language models.

### Canonicalization of Permutation Symmetries

The weights of neural networks have several symmetries, including permutation symmetries. We address these symmetries when training *DeepWeightFlow* by applying two canonicalization methods as preprocessing steps:

**Git Re-Basin**

$$\mathbf{argmax}_{P_\ell}\langle W_\ell^B, P_\ell W_\ell^A P_{\ell-1}^T\rangle + \langle W_{\ell+1}^B, P_{\ell+1} W_{\ell+1}^A P_\ell^T\rangle$$

Git Re-Basin finds permutation matrices $P_\ell$ that bring model $A$'s weights into model $B$'s basin.

**Transfusion: Inter-Head Alignment**

For a single attention head in model $A$, apply singular value decomposition to obtain the projection matrices. For every head in a layer of model $A$, construct a distance and minimize

$$P_{\text{inter-head}} = \text{argmin} \sum D_{I,P[i]}$$

**Transfusion: Inter-Head Alignment**

For a single attention head in model $A$, apply singular value decomposition to obtain the projection matrices. For every head in a layer of model $A$, construct a distance and minimize

$$P_{\text{inter-head}} = \text{argmin} \sum D_{I,P[i]}$$

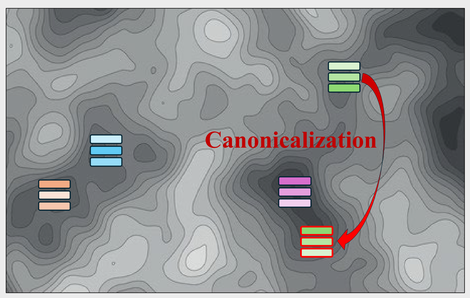



*Figure 1: Visualizing the canonicalization process in model's loss landscape. The green models color gradient becomes aligned with the purple models' color gradient.*

### Dual PCA for Weight Compression

Dual PCA has the following steps

1. Construct the Gram matrix block-wise
2. Randomized Eigenvalue Decomposition
3. Principal Components in Parameter Space

The computational complexity is linear in the weight space dimension

$$\mathcal{O}(n^2 d)$$

### Comparing Model Generation Methods

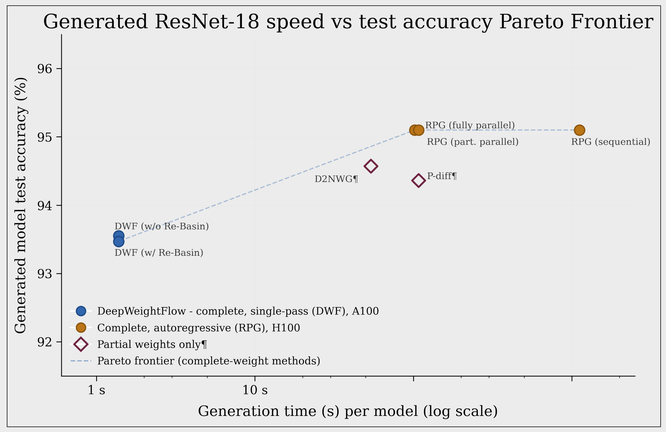
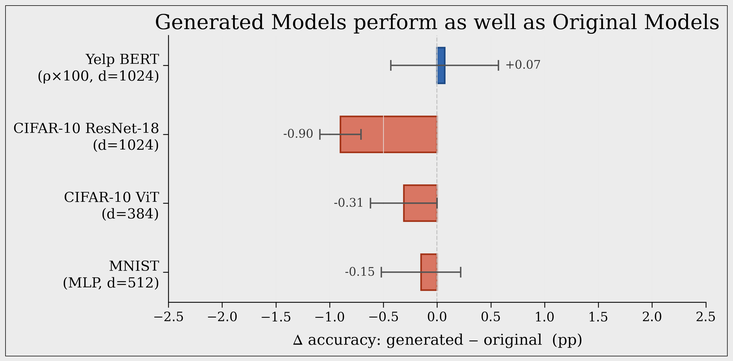



*Figure 2: The generation speed vs accuracy Pareto frontier for generated ResNet-18s on CIFAR-10. The dashed blue line indicates the frontier, where points on the line indicate generated models equal to originals in evaluation. Models only generate subsets of weights. Typically, BatchNorms or decision heads.*

DeepWeightFlow models with and without canonicalization lie on the Pareto frontier, maintaining near equal performance for the generated and original model and up to 150 faster generation than other methods. In contrast to some other methods, we generate all weights.

*DeepWeightFlow* models are faster to train than comparable methods: On ViT-Tiny, RPG reports training times from 6-14 hours across settings, compared to DeepWeightFlow's ca. 20 minutes on ViT-Small-192.

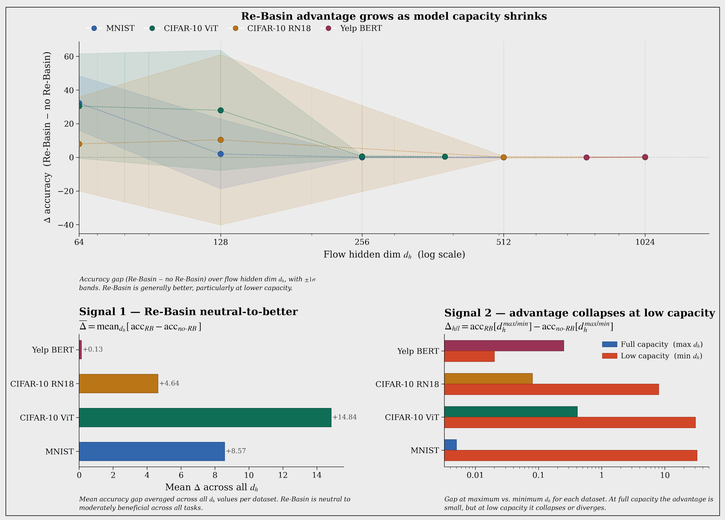
### Generated Models Perform Competitively with Originals




The performance of models generated by *DeepWeightFlow* is nearly indistinguishable from original model, across domains including language modeling with BERT-118M and image classification with ResNet-18s and other models from $\mathcal{O}(100K) - \mathcal{O}(10M)$.

*Figure 3: Models generated by DeepWeightFlow perform at or neat parity with original SGD trained models in evaluation. Uncertainties are propagated with quadrature with N=100 in all cases. The hidden dimension, d, is the optimal selected from our Table 5.*

### Impacts of Canonicalization on Generation Quality




*Figures 5 (top) Test Accuracy increase of models generated with DeepWeightFlow with Git Re-Basined vs a version without canonicalization. Canonicalization of the training set produces better results, but the advantage decreases as model size grows. (lower left) Mean test accuracy increase of models generated using DeepWeightFlow with Git Re-Basin vs a version using no canonicalization for different settings of the hidden dimension $d_h$. Canonicalization leads to better results. (lower right) Test accuracy increase of models generated using DeepWeightFlow with Git Re-Basin vs a version without canonicalization for minimum and maximum hidden dimension $d_h$ settings.*

*DeepWeightFlow* models with lower capacity trained on canonicalized datasets tend to outperform those trained non-canonicalized datasets. The gap closes as model capacity increases.

### Transfer Learning

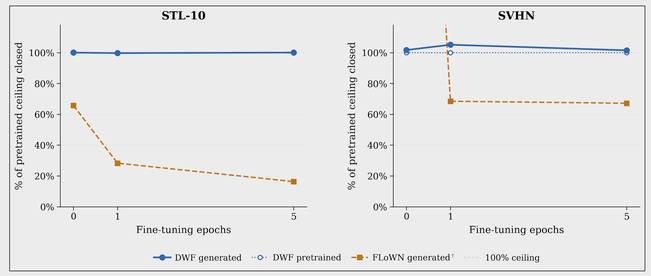



*Figure 6: Transfer learning performances of DeepWeightFlow and FLoWN (Sarigh et. al., 2025) for ResNet-18s transferring from CIFAR-10. DeepWeightFlow generated models more consistently recover near perfect transfer performance. Transfer performance is the gap between random initialization and pretrained CIFAR-10 models closed at 0, 1, and 5 epochs of fine-tuning.*

*DeepWeightFlow* generated weights reach the pretrained ceiling on STL-10 at zero fine-tuning epochs without target task conditioning and remain there through epoch 5. The generated distribution encodes transferable representations, not CIFAR-10 memorization. *DeepWeightFlow* tracks SVHN pretrained accuracy to within noise by epoch 5. *FLoWN*, despite being conditioned on target-task support images at generation time and tasked only with generating batch-norm parameters, closes just ca. 66% of the STL-10 ceiling at epoch 0 and never catches up.

### Diversity and Memorization

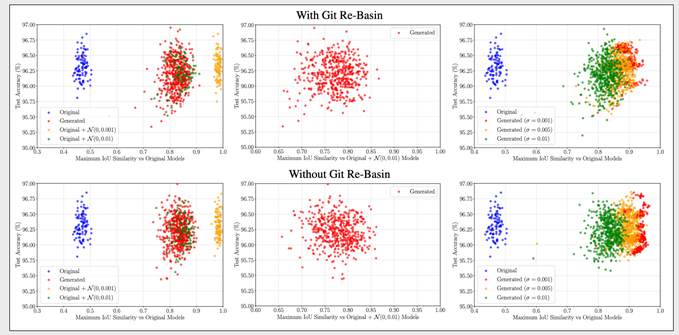



*Figure 6: Comparing IoU (Jaccard) scores of original models, original models with noise, and models generated by DeepWeightFlow. They show that DeepWeightFlow generates distinct weights.*

A common metric of memorization is maximum-Intersection-over-Union (IoU),

$$\text{IoU} = \frac{\left|P_1^{wrong} \cap P_2^{wrong}\right|}{\left|P_1^{wrong} \cup P_2^{wrong}\right|}$$

1. The original models are extremely diverse with respect to each other (IoU $\approx$ 0.5), as shown by the blue clouds in the leftmost column.
2. As expected, adding Gaussian noise produces networks meaningfully different from the original models, $0.8 \lesssim \text{IoU} \lesssim 1.0$.
3. *DeepWeightFlow* generates models that are meaningfully different than original models with added noise, as shown in the center column.

Note that the training datasets for *DeepWeightFlow* consist of 100 independently initialized and trained model checkpoints, not a model zoo or collection of checkpoints from a single parent model.

### Primary References

*Samuel Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. Git re-basin: Merging models modulo permutation symmetries. In The Eleventh International Conference on Learning Representations, 2023.*

*Filippo Rinaldi, Giacomo Capitani, Lorenzo Bonicelli, Donato Crisostomi, Federico Bolelli, ELISA FICARRA, Emanuele*

*Rodola, Simone Calderara, and Angelo Porrello. Update your transformer to the latest release: Re-basin of task vectors. In Forty-second International Conference on Machine Learning, 2025*

*Daniel Saragih, Deyu Cao, Tejas Balaji, and Ashwin Santhosh. Flow to learn: Flow matching on neural network parameters.*

*In Workshop on Neural Network Weights as a New Data Modality, 2025b Kai Wang, Zhaopan Xu, Yukun Zhou, Zelin Zang, Trevor Darrell, Zhuang Liu, and Yang You. Neural Network Diffusion,*

*February 2024 Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In The Eleventh International Conference on Learning Representations, 2023*