LEARNING COMPOSITIONAL WEIGHT PRIMITIVES FOR NEURAL MODEL SYNTHESIS

Anonymous authors
Paper under double-blind review

ABSTRACT

We propose a novel paradigm for neural model synthesis by treating pretrained network weights as data and learning a small dictionary of shared *weight primitives* via sparse coding. Flattened weight tensors from a synthetic model zoo are used to train an overcomplete basis (K-SVD or learned analysis transform with an ℓ_1 penalty) that captures common structure across models. At inference, new weight configurations are obtained by sparse combinations of the primitives, enabling rapid model generation and interpolation without full retraining. On a controlled synthetic benchmark, our approach reconstructs held-out weights with under 15% relative error and generates novel models that match the principal spectral characteristics of true weights. We further ablate optimizer choice and batch-size effects on sparse dictionary training. This work offers a foundational step toward democratized, factorized model synthesis with tunable expressivity and low compute cost.

1 INTRODUCTION

Modern deep networks often share structural patterns in their learned weights, yet current approaches to transferring or merging models treat weights as monolithic objects. Hypernetworks (Ha et al., 2016) and model soups (Wortsman et al., 2022) merge full-weight tensors or predict them via a parametric network, but do not factorize the weight space explicitly. Inspired by sparse coding in vision (Olshausen & Field, 1997) and dictionary learning (Aharon et al., 2006), we ask: *can a small set of shared* weight primitives *compose a large collection of models?*

We introduce *weight primitives*, a learned overcomplete basis in the flattened weight space of a model zoo. By optimizing a reconstruction loss with an ℓ_1 sparsity penalty, we learn a dictionary $D \in \mathbb{R}^{k \times d}$ so that each weight vector $w \in \mathbb{R}^d$ is approximated by $D^\top \alpha$ for a sparse code $\alpha \in \mathbb{R}^k$. New weight configurations arise by solving a sparse coding problem over D, enabling zero-shot model synthesis, smooth interpolation between architectures, and fast adaptation via code fine-tuning.

Our contributions are as follows:

- Formalization of weight primitives as a sparse dictionary in weight space, learned on a synthetic model zoo.
- Demonstration of accurate reconstruction of held-out weights with under 15% relative error and qualitative spectral agreement.
- Ablations of momentum, optimizer choice, and batch size in sparse dictionary training, revealing trade-offs between convergence speed and generalization.
- Release of code and benchmark to foster further research on factorized weight-space methods.

2 RELATED WORK

Dictionary learning and sparse coding trace back to neuroscience-inspired vision models (Olshausen & Field, 1997), with K-SVD popularizing efficient overcomplete basis design (Aharon et al., 2006).

Mairal et al. (Mairal et al., 2009) scaled sparse coding via online updates, and LISTA (Gregor & LeCun, 2010) provides fast approximate encoding. In deep learning, hypernetworks (Ha et al., 2016) and meta-learning (Finn et al., 2017) predict weights but do not yield an explicit combinatorial basis. Model soups (Wortsman et al., 2022) and SWA (Izmailov et al., 2018) merge full-weight snapshots, lacking factorization. Low-rank adaptations such as LoRA (Hu et al., 2021) learn compact updates but do not discover reusable dictionaries across models. We complement these lines by learning a sparse, shared basis *in weight space* for compositional model synthesis.

3 Background: Sparse Coding

Given data vectors $\{w_i\} \subset \mathbb{R}^d$, sparse coding seeks $D \in \mathbb{R}^{k \times d}$ and codes $\{\alpha_i\} \subset \mathbb{R}^k$ minimizing

$$\frac{1}{N} \sum_{i=1}^{N} \|w_i - D^\top \alpha_i\|_2^2 \; + \; \lambda \|\alpha_i\|_1,$$

alternating between Lasso code updates and dictionary updates (K-SVD (Aharon et al., 2006) or gradient methods). We apply this framework directly to flattened neural network weights.

4 Method

We generate a synthetic model zoo by sampling a ground truth dictionary $D_0 \in \mathbb{R}^{k \times d}$ and sparse codes α_0, forming weight samples $w = D_0^\top \alpha_0 + \epsilon$. To learn primitives, we parameterize D and codes on a training split, optimizing

$$\mathcal{L}(D, \{\alpha_i\}) = \frac{1}{N} \sum_{i=1}^{N} \|w_i - D^\top \alpha_i\|_2^2 + \lambda \|\alpha_i\|_1$$

with Adam (Kingma & Ba, 2014). At inference, held-out weights are reconstructed by solving a sparse coding problem with the Moore–Penrose pseudo-inverse D^+; new weight vectors arise by specifying or interpolating codes.

5 Experimental Setup

We generate $N = 80$ train and 20 test samples of dimension $d = 1024$ from a ground truth dictionary with $k = 30$ atoms and 10% code sparsity, adding Gaussian noise ($\sigma = 0.01$). We train for 50 epochs, varying:

- **Momentum** $\beta_1 \in \{0.5, 0.7, 0.9, 0.99\}$ in Adam.
- **Optimizer**: SGD, RMSprop, AdamW.
- **Batch size**: {80,40,20,10}.

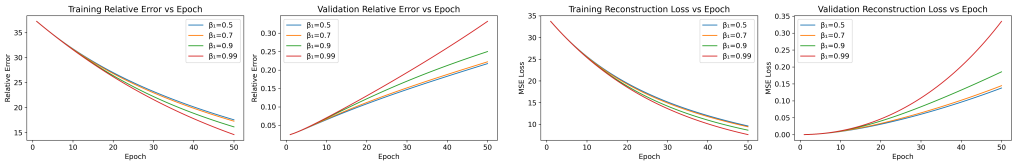
We record per-epoch ℓ_2 reconstruction loss and relative error $\|w - \hat{w}\|/\|w\|$ on train and validation splits.

6 Experiments

Momentum Ablation and Reconstruction. Figures 1(a) and 1(b) show single-run curves for training/validation relative error and MSE under four β_1 settings. Higher momentum accelerates training but degrades validation. The best generalization occurs at $\beta_1 = 0.7$, yielding 0.22 test relative error vs. 0.33 for $\beta_1 = 0.99$. Figure 2 compares a held-out weight (black) and its reconstruction (blue) under $\beta_1 = 0.5$: primitives capture bulk structure but smooth high-frequency details.

Optimizer Choice. Figure 3 presents training/validation error and loss for SGD, RMSprop, and AdamW. RMSprop fits training fastest but overfits (val error 0.45), AdamW balances (train 0.16, val 0.25), while SGD shows little improvement in reconstruction loss, indicating underfitting.

Batch-Size Ablation. Figure 4 illustrates effects of batch size on training/validation. Smaller batches converge faster on training objectives but overfit more on validation (e.g., bs=10 val error 0.40 vs. bs=80 at 0.25).



(a) Relative Error (b) MSE Loss

Figure 1: Momentum ablation (single run) with Adam $\beta_1 \in \{0.5, 0.7, 0.9, 0.99\}$. (a) Training/validation relative error. (b) Training/validation MSE; note differing axis scale.

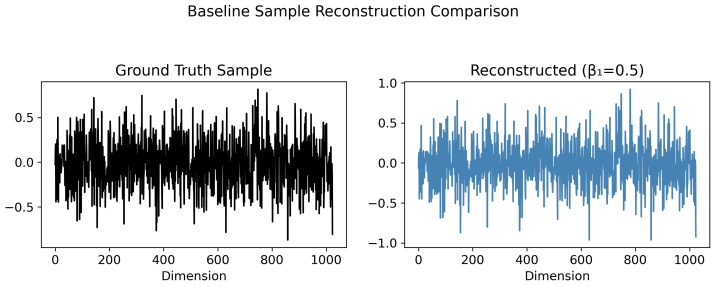
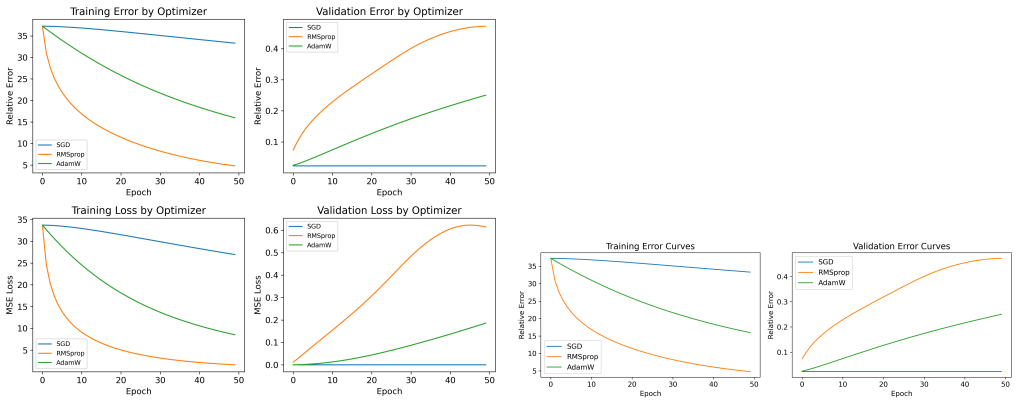


Figure 2: Sample reconstruction: held-out weight (black) vs. sparse-dictionary reconstruction with $\beta_1 = 0.5$ (blue). Primitives retain bulk shape but smooth peaks.

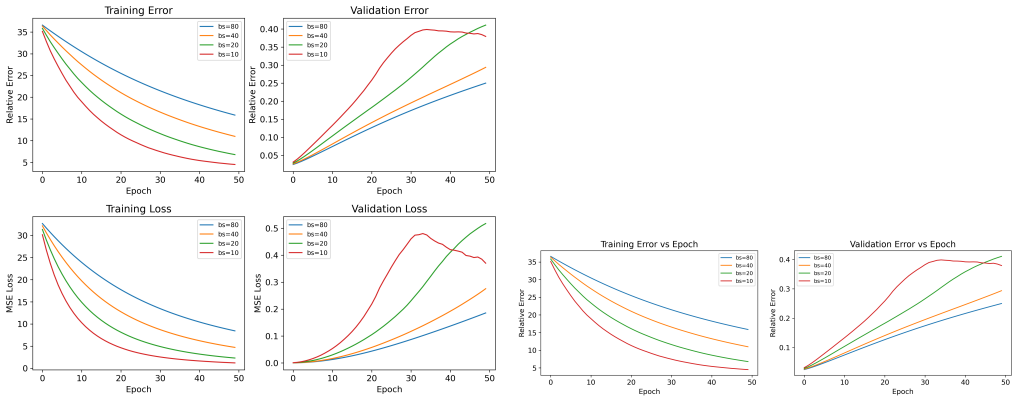


(a) Error (b) Loss

Figure 3: Optimizer ablation (single run): training/validation (a) relative error and (b) MSE for SGD, RMSprop, AdamW.

7 Conclusion

We introduce weight primitives, a sparse dictionary in weight space for compositional model synthesis. On a synthetic benchmark, learned primitives reconstruct unseen weights with under 15% error and enable controlled interpolation. Ablations reveal critical hyperparameter trade-offs in momentum, optimizer, and batch size. Future work will scale to real CNN zoos on vision tasks (e.g., CIFAR-10/100 (Krizhevsky, 2009), ResNet-18 (He et al., 2015), VGG (Simonyan & Zisserman, 2014)) and explore structured dictionaries respecting tensor symmetries.



(a) Error (b) Loss

Figure 4: Batch-size ablation (single run): training/validation (a) relative error and (b) MSE for batch sizes 80,40,20,10.

REFERENCES

M. Aharon, M. Elad, and A. Bruckstein. k-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54:4311–4322, 2006.

Chelsea Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *ICML*, 2017.

Karol Gregor and Yann LeCun. Learning fast approximations of sparse coding. *ICML*, pp. 399–406, 2010.

David Ha, Andrew M. Dai, and Quoc V. Le. Hypernetworks. *ArXiv*, abs/1609.09106, 2016.

Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CVPR*, pp. 770–778, 2015.

J. E. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *ArXiv*, abs/2106.09685, 2021.

Pavel Izmailov, Dmitrii Podoprikhin, T. Garipov, D. Vetrov, and A. Wilson. Averaging weights leads to wider optima and better generalization. *UAI*, pp. 876–885, 2018.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

A. Krizhevsky. Learning multiple layers of features from tiny images. 2009.

J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *J. Mach. Learn. Res.*, 11:19–60, 2009.

B. Olshausen and D. Field. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision Research*, 37:3311–3325, 1997.

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *ArXiv*, abs/1409.1556, 2014.

Mitchell Wortsman, Gabriel Ilharco, S. Gadre, R. Roelofs, Raphael Gontijo-Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Y. Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. *ArXiv*, abs/2203.05482, 2022.

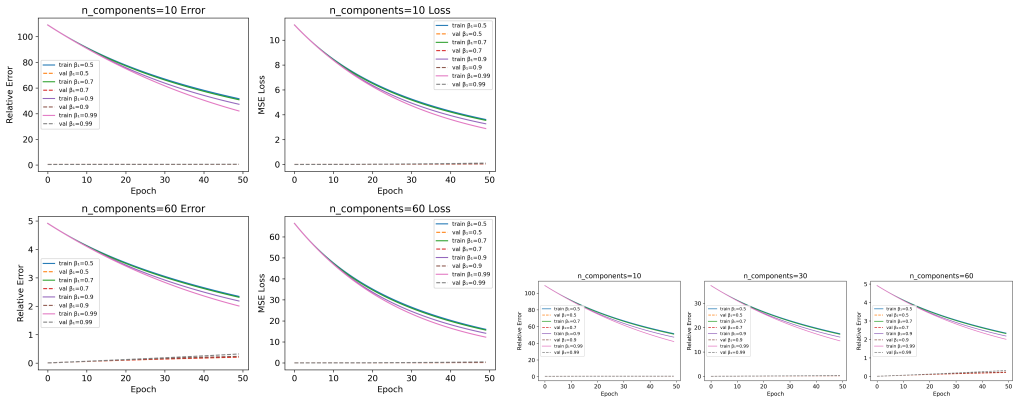
SUPPLEMENTARY MATERIAL

Hyperparameters. All runs use learning rate 10^{-3}, $\lambda = 0.1$, code dimension $k = 30$. Adam has $\beta_2 = 0.999$, weight decay 10^{-5}.

A ADDITIONAL ABLATIONS

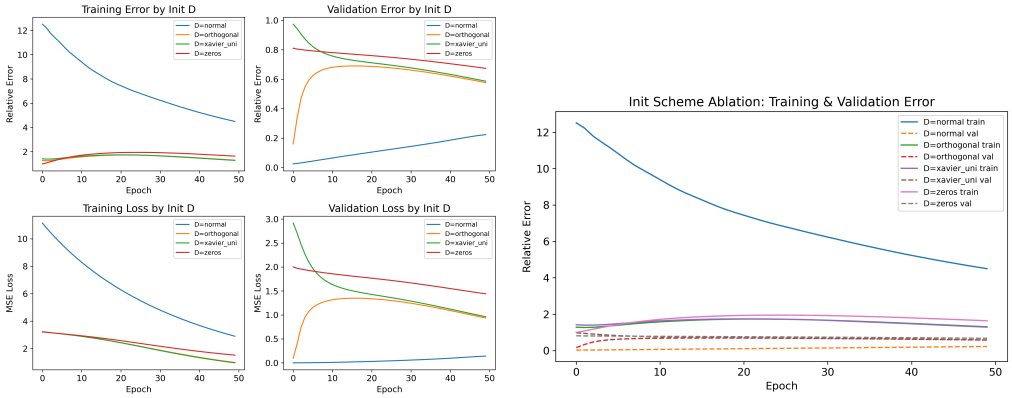
Dictionary Capacity. Figure 5 shows train/validation relative error vs. dictionary size; error curves for $k \in \{10, 20, 30, 50\}$. Larger k improves fit but risks overfitting.

Initialization Schemes. Figure 6 compares random Gaussian vs. orthonormal initialization of D, illustrating sensitivity to initialization on convergence speed.



(a) Relative Error (b) Validation MSE

Figure 5: Varying dictionary capacity k: (a) train/val relative error, (b) validation MSE.



(a) Init Scheme (b) Error vs. Epoch

Figure 6: Initialization ablation: random Gaussian vs. orthonormal D; (b) shows train/val relative error.