Neural Network Training as Key Search Across Resolution Bands

Jurij Jukić
Independent Researcher
Zagreb, Croatia
j.jjukicc@gmail.com

Abstract

Scalar loss in machine learning obscures what a neural network learns and when. While the specific circuits in a network are opaque, we can observe how a network learns across different resolutions of data. Our hope is that our resolution-based approach makes theories of learning easier to formulate and test. We illustrate this by proposing a phenomenological model, key search, with a concrete methodology that decomposes the learning process across three axes: parameter time, resolution band, and band loss. Parameter time is a reparameterization of training time by distance traveled along the optimizer's path in parameter space. A resolution band is the content present at a higher resolution level but absent at a lower one, obtained as an orthogonal component of a multiresolution analysis. Band loss is the loss restricted to that band, rather than the full resolution. Key search predicts three results, which we test on 12 CelebA 64×64 autoencoding runs (6 CNN, 6 ViT): 1. the gap between a band's current loss and its floor is well fit by a Weibull curve in parameter time over the active window ($R^2 > 0.99$ for bands 2–64); 2. band activation is ordered across scales, with finer bands activating later (Spearman $\rho = 0.79$), decaying over wider parameter time windows ($\rho = 0.96$), reaching higher floors ($\rho = 0.96$), and starting with smaller removable loss ($\rho = -0.89$); 3. the ordinary scalar loss, obtained by summing band residuals, is also well summarized by a single Weibull curve in parameter time (median $R^2 = 0.996$, median $\Delta\mathrm{BIC} = -28.83$ versus exponential).

1 Introduction

A neural network's training loss is a single scalar that hides everything about what the network is learning and when. This paper asks whether decomposing that scalar into interpretable components can reveal structure in the learning process that the aggregate curve obscures.

We study image autoencoders, where a natural decomposition is available: the Haar wavelet basis splits an image into orthogonal resolution bands, each capturing the content present at one spatial scale but absent at the next coarser one. Because the bands are orthogonal and the loss is squared error, band losses sum exactly to total loss. This lets us track how much of the learning at any point in training is happening at each resolution, rather than observing only the aggregate.

Alongside this decomposition, we introduce a second methodological choice: measuring training progress not in raw step count but in parameter time — the cumulative Euclidean distance traveled by the parameters. The intuition is that parameter distance tracks how far the optimizer has actually searched, whereas raw step count confounds search progress with the optimizer's varying step size.

With these two tools — band decomposition and parameter time — we propose and test a phenomenological model we call key search. The idea is that training searches for keys: parameter configurations that unlock loss reduction at a given resolution band. Short keys are easy to find and tend to be

Preprint.

effective (coarse structure accounts for large chunks of loss); long keys take more search and yield smaller returns (fine detail is particular and accounts for less loss). From this picture, three predictions follow:

1. **Shape.** Each band's residual — the difference between current loss and final loss — follows a Weibull decay in parameter time, reflecting a search process whose rate changes over time.
2. **Ordering.** Coarse bands activate earlier, achieve their final loss sooner, reach lower floors, and have more total removable loss than fine bands.
3. **Full-loss residual.** The full-loss scalar residual, which is the sum of the band residuals, is also well described by a single Weibull curve in parameter time.

We test these predictions on 12 CelebA 64×64 autoencoding runs (6 CNN, 6 ViT). All three are supported in this setting: per-band Weibull fits achieve $R^2 > 0.99$ for bands 2–64; ordering across bands holds with Spearman ρ between 0.79 and 0.96 on all four measured quantities; and the full-loss Weibull fit achieves median $R^2 = 0.996$ with $\Delta\mathrm{BIC} = -28.83$ versus the exponential special case.

Our hope is that the methodology — band decomposition plus parameter time — is useful independently of the specific theory tested here. Decomposing loss into orthogonal components, each with its own trajectory, provides a richer set of observables against which theories can be checked.

2 Related work

Scaling laws. Kaplan et al. [2020] established that language model loss follows power laws in model size, dataset size, and compute; Hoffmann et al. [2022] revised the compute-optimal tradeoff. Subsequent work has sought theoretical explanations for these power-law exponents [Bahri et al., 2024, Sharma and Kaplan, 2022]. These analyses fit a single scalar loss against raw training step or compute. Our work differs in two respects: we decompose loss by resolution band before fitting, and we reparameterize training time as cumulative parameter-space distance rather than step count. The resulting per-band curves are Weibull rather than power law.

Spectral bias and the frequency principle. Rahaman et al. [2019] and Xu et al. [2020] independently showed that neural networks learn low-frequency components of the target function before high-frequency ones, a phenomenon called spectral bias or the frequency principle (F-Principle). Our band-ordering result (Result 2) is a counterpart of this observation on images analyzed with Haar wavelets: coarse bands are learned before fine bands. However, our focus is different. The spectral bias literature characterizes *which* frequencies are learned first; we additionally measure *how much loss* each band contributes, fit a parametric decay curve to the per-band dynamics, and connect the ordering to a theoretical framework (key search).

Bennett's logical depth. Bennett [1988] defined the logical depth of an object in terms of the time required by a near-shortest program to produce it, formalizing the distinction between objects that are merely information-rich and those that are computationally expensive to produce. Hoang and Guerraoui [2018] conjectured that deep learning succeeds because real-world data has large non-parallelizable logical depth, connecting Bennett's notion to network depth (number of layers). Our use of logical depth is different: we use it to motivate a model of training *dynamics*, where depth corresponds to the parameter-time of finding a loss-reducing key, not to architectural depth.

Parameter-space geometry. The geometry of training trajectories has been studied through loss landscape visualization [Li et al., 2018] and the low-dimensional structure of gradient updates [Gur-Ari et al., 2018]. Our parameter time $s(t)$ — cumulative Euclidean distance in parameter space — is related in spirit to these geometric views, but we use it as a reparameterized training time under which per-band losses become clean Weibull curves. To our knowledge, cumulative parameter distance has not previously been used as a time axis for fitting training dynamics.

Learning curve functional forms. In the machine learning scaling law literature, power laws are the dominant functional form [Kaplan et al., 2020, Hoffmann et al., 2022]. We are not aware of prior work fitting Weibull curves to neural network training loss, though the Weibull distribution appears in survival analysis and reliability engineering.

3 Setup

Resolution. We use *resolution* to refer to spatial scale in a multiresolution decomposition. A resolution level r corresponds to an $r \times r$ spatial grid. We work with dyadic levels $r \in \{1, 2, 4, 8, 16, 32, 64\}$ for our CelebA 64×64 images.

Resolution band. For an image at resolution $N \times N$, a resolution band captures the content added at a specific spatial scale. Concretely, we use the Haar decomposition to split an image into nested resolutions: the coarsest level captures global block averages, each finer level captures the detail added relative to the coarser one. For a 64×64 image, this yields 7 bands at dyadic resolutions (1, 2, 4, 8, 16, 32, 64), each orthogonal to the others.

Band loss. The loss a model achieves at a given resolution band. In general, band losses need not sum to total loss; whether they do depends on the loss function and the decomposition. When the bands are orthogonal (the content at each finer band is orthogonal to all content at coarser ones) and the loss is squared-error, the band losses sum exactly to the total loss. For an input image x and model reconstruction $f(x; \theta_t)$ at training time t,

$$\mathrm{MSE}(x, f(x; \theta_t)) = \sum_i L_i(x, f(x; \theta_t)). \tag{1}$$

For this reason, we restrict our empirical study to image reconstruction with MSE loss. Formal definitions and derivation are given in Appendix A.6.

Floor. For each band i, we denote its floor, the final achieved loss as L_i^∞ (estimated from saturated late-training behavior).

Residual. We then define residual-to-floor as

$$R_i(t) = L_i(t) - L_i^\infty, \tag{2}$$

This is the amount of removable loss still remaining in band i at time t, decaying to zero as training saturates.

Because the Haar bands are orthogonal and the loss is MSE, the ordinary full-resolution loss is the sum of the band losses. We therefore define the full-loss residual, also called the total residual, as

$$R_{\mathrm{tot}}(t) = \sum_i R_i(t) = \sum_i \left(L_i(t) - L_i^\infty\right).$$

Equivalently,

$$R_{\mathrm{tot}}(t) = L_{\mathrm{tot}}(t) - L_{\mathrm{tot}}^\infty,$$

where $L_{\mathrm{tot}}^\infty = \sum_i L_i^\infty$.

Raw time. We use t to denote the training step count.

Parameter time. Parameter-space distance measures how much the model's parameters have moved. We define parameter time $s(t)$ as the cumulative parameter-space distance along the optimizer's trajectory up to raw training step t:

$$s(t) = \sum_{k=1}^{t} \|\theta_k - \theta_{k-1}\|,$$

where θ_k denotes the model's parameters at step k.

Empirically, the relationship between parameter time and raw time t is approximately power-like:

$$s(t) \approx a + bt^\gamma,$$

with median $\gamma = 0.368$, run-to-run SD $= 0.008$, and median full-window $R^2 = 0.999$. More in Appendix A.7.

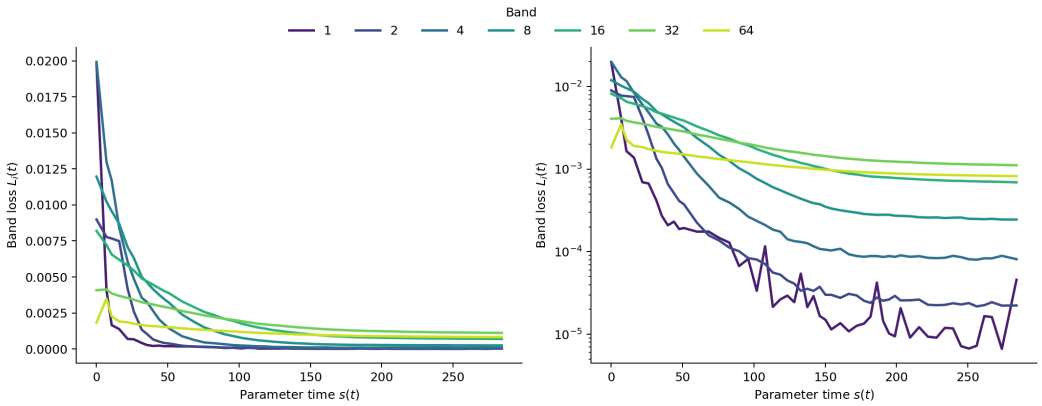

Figure 1: Band loss L_i over parameter time s for one CelebA ViT autoencoding run, with each line corresponding to one resolution band. Left: linear y-axis. Right: log y-axis.

4 Theory

4.1 Key search

This section motivates the theoretical constructs we use and informally introduces the main quantities we measure.

Bennett's logical depth formalizes the intuition that some structures are not merely information-rich but computationally expensive to produce. He compares two quantities: a string's Kolmogorov complexity is the length of the shortest program that generates it, and its depth is the runtime of that program. A random string has high K complexity (no short program exists) but low depth (it is copied verbatim). The digits of π have low K complexity (a short formula generates them) but high depth (the formula takes long to run).

Bennett's insight connects depth to search cost: short programs of length ℓ are rare among the 2^ℓ candidates of that length, and so take long to find by enumeration. A deep string is one that cannot be produced without finding such rare, hard-to-find programs.

We adopt this framing for neural network training and postulate that training searches for keys: programs that reduce loss at a given resolution band. For each key, we distinguish its depth (the parameter time required to find it) from its effectiveness (the loss reduction once found). From this idea, three predictions follow.

Prediction 1 (shape). The search process itself has a characteristic shape. In a simple search for a key of length ℓ, the per-step probability of finding it is roughly $2^{-\ell}$. Assuming independent trials, the probability that the key has not yet been found after t steps is $(1 - 2^{-\ell})^t$, which decays exponentially in t. We relate this remaining probability to the residual loss $R_i(t)$. However, real training is not a simple search. Keys roughly correspond to parameter configurations, and gradient descent follows the loss landscape: keys in steep, easily-descended regions are reached first; those in flatter or more distant regions take longer. Therefore, the rate at which keys are found need not remain constant over time, and the residual need not decay at a constant rate. The resulting residual is not expected to be a pure exponential. We model it with the Weibull family, a generalization of the exponential that allows the decay rate to speed up or slow down over time.

These ideas also motivate our choice of parameter time. Parameter time reflects how far the search has actually traveled in parameter space, i.e. different keys or parameter configurations that have been tried, which we presume tracks search progress more accurately than raw training step count.

Prediction 2 (ordering). Across bands, depth and effectiveness tend to inversely correlate. Shallow keys tend to be more effective. For example, at 2×2 or 4×4, a short program capturing overall face structure reduces the band's loss by a large amount. Deep keys tend to be less effective, since fine-scale structure is more particular and accounts for a smaller chunk of total loss. This gives a

pattern across bands: coarse bands begin learning early (shortest effective key is shorter than for fine bands) and reduce loss by a large margin (more removable loss than fine bands).

Prediction 3 (full-loss residual). The same key-search logic that applies to each band applies to the full loss: the aggregate training process is also a search over keys, so the full-loss residual should also follow a Weibull decay in parameter time. Moreover, because the band losses are additive, the ordinary scalar residual is the sum of the band residuals. The Weibull family is not closed under summation, so a sum of per-band Weibulls need not itself be Weibull-shaped. Nevertheless, we find that a single Weibull fits the full-loss residual well (Result 3).

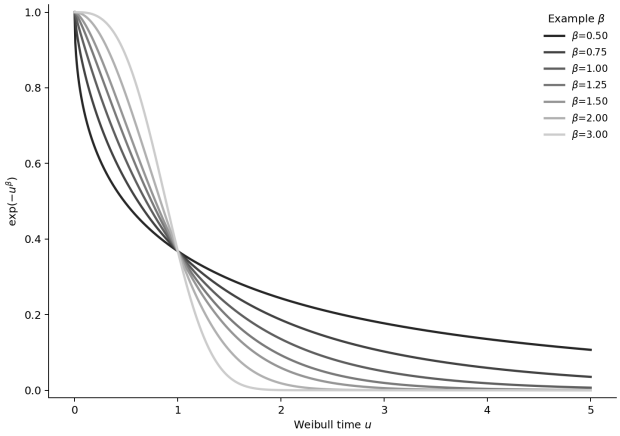

Figure 2: To illustrate how the Weibull β parameter shapes the decay curve, this figure shows stylized Weibull curves for different β values (schematic, not a fit to data). For $\beta = 1$, the curve is a pure exponential. For $\beta < 1$, decay is fast initially then slows (diminishing returns). For $\beta > 1$, decay is initially slow — a "warmup" phase — followed by faster decay.

4.2 Residual as a Weibull tail

We refer to Prediction 1 from the key search section, where we postulate that the residual follows a Weibull-shaped tail. We define the band's tail T_i as

$$T_i(x) = A_i^{\text{tail}} \exp\big(-(x/\omega_i)^{\beta_i}\big) \quad \text{for } x \geq 0, \tag{3}$$

where A_i^{tail} is the theoretical tail amplitude, ω_i is a scale parameter, and β_i is the Weibull shape exponent. The argument x is not training time itself, it represents search progress or frontier position along the tail.

Let $x_i(t)$ denote the frontier position for band i. Then the band's residual is

$$R_i(t) = T_i(x_i(t)).$$

The frontier x_i depends on the band's data, the learner, and training time — different learners applied to the same data would progress along the tail at different rates. This separates the tail (a property of the data) from the frontier (how the specific learner moves along it).

The frontier x_i is latent and not directly measured. We therefore cannot directly observe the Weibull shape in x-space. What we can observe is R_i as a function of parameter time s.

The separation between tail and frontier is a conceptual clarification, not an empirical decomposition. The frontier x_i is latent and not directly measured; in this paper we fit Weibull curves directly to $R_i(s)$ and do not attempt to recover x_i. If the frontier is a monotonic power-like function of s, then a Weibull in x manifests as a Weibull in s. We find that it does: the fitted Weibull describes band residuals well (Result 1). Whether parameter time is a good proxy for frontier position across

architectures and datasets remains an open question; we observe only that it produces clean fits for our CNN and ViT runs on CelebA.

5 Experimental Setup and Measurement Protocol

We study training dynamics in image autoencoding on CelebA (64×64). We train 12 runs total: 6 CNN and 6 ViT, using seeds 45–50 for each architecture. The CNN is a 4-layer strided convolutional autoencoder (1.83M parameters); the ViT uses 4 transformer layers with 4×4 patches (2.78M parameters). Both use a 128-dimensional latent and Sigmoid output. Training uses plain Adam (lr=3×10^{-4}, batch size 128) for 50 epochs, with no learning-rate schedule, no weight decay, and no gradient clipping. Band losses $L_i(s)$ are computed on a fixed 256-image test panel per run.

Parameter time $s(t)$ is the cumulative sum of Euclidean distances between successive saved checkpoints. The residual is $R_i(s) = L_i(s) - L_i^\infty$, where the floor L_i^∞ is estimated as the mean band loss over the last five saved checkpoints. Appendix A.9 checks that replacing the average by the final checkpoint gives nearly identical floors.

5.1 Threshold crossings

Our measurements are available only at a finite set of saved checkpoints, indexed by k, with parameter times s_k. For band i, define the normalized residual at checkpoint k as

$$q_i(s_k) = \frac{R_i(s_k)}{R_i(s_0)},$$

where s_0 denotes the initial checkpoint.

For threshold $\alpha \in (0, 1)$, the α-crossing $c_i(\alpha)$ is the parameter-time location at which q_i first falls below α, estimated by linear interpolation between the two adjacent saved checkpoints that bracket the threshold. The interpolation formula is given in Appendix A.10. Crossings are used for two purposes.

Band ordering. Crossings define the quantities that describe band-ordering: onset is the 0.95 crossing, i.e. the parameter time at which the band has achieved the first 5% of its loss reduction:

$$\tau_i = c_i(0.95),$$

and active width is the distance between the 0.95 and 0.01 crossings, the length of time during which the band has achieved most of its loss reduction:

$$w_i = c_i(0.01) - c_i(0.95).$$

Active windows for Weibull fits. For each per-band fit, the active window begins at the first saved checkpoint after $c_i(0.95)$ and ends at the last saved checkpoint before $c_i(0.01)$. This threshold pair is fixed before fitting rather than chosen to optimize fit quality. Thus the crossings define the window boundaries, but the fitted data are only the saved checkpoint observations inside that window. Direct global fits are single Weibull fits to the full-loss residual R_{tot}. Full-loss fits do not use crossing-defined active windows; they use the full saved-checkpoint range.

5.2 Fit quality

Reported R^2 values are computed directly on the residual values $R_i(s)$ in the same linear scale used by MSE, not on the log scale.

For model comparison we report

$$\Delta\text{BIC} = \text{BIC}_{\text{Weibull}} - \text{BIC}_{\text{exp}},$$

which penalizes the Weibull's extra shape parameter β unless it earns its keep through better fit. Negative values favor Weibull over the exponential special case ($\beta = 1$).

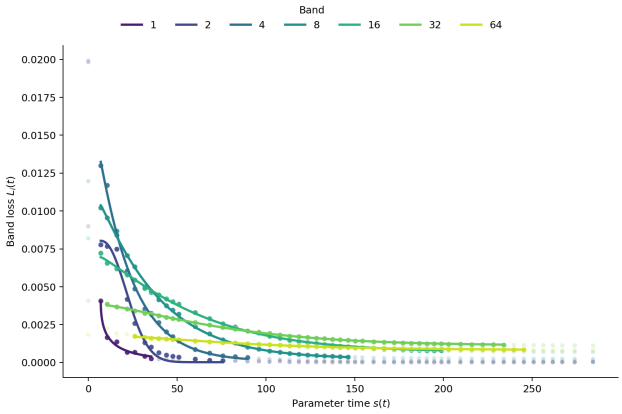

Figure 3: Weibull band fits for one representative run. Dots are pale outside the active window.

Full architecture specifications, checkpoint schedule, crossing interpolation formula, fit procedure, and robustness analyses are given in Appendices A and B.

6 Results

6.1 Result 1: Weibull

Figure 1 shows that band trajectories do not necessarily decay smoothly from the start of training. Band 64 rises above its initial value before decaying; band 32 shows a small early bump; coarse bands such as band 2 drop almost immediately. In the key-search framing, this corresponds to a pre-decay phase before the optimizer has reached a key that produces useful loss reduction in that band.

For this reason, we do not fit Weibulls to the full per-band trajectory. Instead, for each band we fit only the checkpoint observations inside the active window defined in Section 5: the first saved checkpoint after the 0.95 crossing through the last saved checkpoint before the 0.01 crossing. No interpolated residual values are used in the fit.

We fit the Weibull tail form of Eq. (3) to each band's residual trajectory $R_i(s)$ using nonlinear least squares with bounds on each parameter.

We obtain $R^2 > 0.99$ across bands 2–64, with band 1 at $R^2 = 0.982$ (Appendix B.2). To assess whether the additional Weibull shape parameter is warranted, we compare Weibull fits against the pure exponential special case $\beta = 1$ via BIC. Weibull is preferred with median $\Delta\text{BIC} = -19.65$ (72/84 fits); under a conservative $n_{\text{eff}} = n/10$ deflation accounting for hidden correlations, Weibull preference still holds at median $\Delta\text{BIC} = -1.73$ (78/84 fits). Fits are performed in parameter time. The active-window choice is not critical: under alternative 0.90/0.10 and 0.80/0.20 threshold windows, fit quality remains stable and the band ordering correlations (shown in the following section) are essentially unchanged (Appendix B.3).

6.2 Result 2: Band ordering

We measure each band's onset $\tau_i = c_i(0.95)$, active width $w_i = c_i(0.01) - c_i(0.95)$, initial residual $R_i(s_0)$, and floor L_i^∞ from its residual trajectory $R_i(s)$ (definitions in Section 5). These crossing-based quantities are used for descriptive ordering; they are not fitted parameters.

1. **Onset** τ_i: finer bands activate later.
2. **Active width** w_i: finer bands decay over a wider s window.
3. **Floor** L_i^∞: finer bands plateau at higher asymptotic loss (i.e. more unresolved loss).
4. **Initial residual** $R_i(s_0)$: finer bands start with less loss available to resolve.

Across the 12 runs, the run-level correlation summaries are:

Descriptor	Median ρ	IQR	Min–max
Onset τ_i	0.79	[0.75, 0.79]	[0.75, 0.89]
Width w_i	0.96	[0.96, 0.96]	[0.93, 1.00]
Floor L_i^∞	0.96	[0.96, 0.96]	[0.96, 0.96]
Initial residual $R_i(s_0)$	-0.89	$[-0.89, -0.88]$	$[-0.89, -0.82]$

6.3 Result 3: Full-loss Weibull

Because the Haar bands are orthogonal and the loss is MSE, the ordinary full-loss residual is the sum of the band residuals:

$$R_{\text{tot}}(s_k) = \sum_i R_i(s_k).$$

We evaluate two quantities.

First, we evaluate each per-band Weibull fit $\widehat{R}_i$ on the full checkpoint grid and sum them to obtain the band-fit reconstruction $\widehat{R}_\Sigma(s_k) = \sum_i \widehat{R}_i(s_k)$. This matches R_{tot} with median $R^2 = 0.986$.

Second, we fit a single Weibull directly to R_{tot} over the full checkpoint range, obtaining median $R^2 = 0.996$ and median $\Delta\text{BIC} = -28.83$ versus the exponential special case. As noted in Prediction 3, the Weibull family is not closed under summation, so this global fit is an empirical regularity rather than a consequence of the bandwise fits.

The direct global fit splits by architecture: ViT is decisively stretched ($\beta = 0.79$, $\Delta\text{BIC} = -73.83$), while CNN is a near-tie with exponential ($\beta \approx 0.95$, $\Delta\text{BIC} \approx +0.95$). At the per-band level both architectures show $\beta > 1$ (bands 4–64), but CNN's sum of bands composes to a shape closer to exponential while ViT's composes to a stretched one.

7 Discussion

We have proposed a methodology for studying training dynamics — decomposing loss by resolution band and reparameterizing time as cumulative parameter distance — and used it to test key search, a phenomenological model of training dynamics. The three predictions of key search are supported by our 12-run CelebA training runs: per-band residuals follow Weibull curves in parameter time, bands activate in a coarse-to-fine order with predictable relationships between onset, width, floor, and initial residual, and the full-loss residual is itself well described by a single Weibull.

The Weibull fits characterize the shape of per-band learning curves but do not explain why training produces this shape. Key search provides one interpretive frame; the empirical regularities may admit others. We do not know which observations are consequences of the data, the architecture, the optimizer, or their interaction: the CNN and ViT runs share dataset and optimizer but differ in architecture, and while per-band Weibull fits hold for both, the global fit shape differs ($\beta < 1$ only for ViT). We study a single dataset (CelebA 64×64), a single task (autoencoding), and small models (1.8–2.8M parameters) trained for 50 epochs; a thorough ablation has not been performed.

The band-loss additivity that makes our decomposition exact relies on two properties: the Haar bands are orthogonal and the loss is squared error. For cross-entropy loss — the standard in language modeling and classification — no obvious analogous Pythagorean decomposition exists, and it is unclear whether language admits a natural orthogonal decomposition comparable to spatial resolution bands for images. Extending this methodology to cross-entropy settings therefore remains an open problem, not just an engineering challenge.

Several directions for future work follow naturally. First, systematic ablations: varying the dataset (beyond CelebA, beyond faces), the architecture (depth, width, convolutional vs. transformer vs. MLP), and optimizer settings (learning rate, schedule, weight decay, SGD vs. Adam) to disentangle which aspects of the observed dynamics are data-driven, architecture-driven, or optimizer-driven. Second, a more careful comparison of parameter time to raw time, including how the equation $s(t) \approx a + bt^\gamma$ varies across settings and whether parameter-time Weibull fits can be related to existing raw-time scaling laws [Kaplan et al., 2020, Hoffmann et al., 2022]. Third, exploring whether "keys" in the key-search framework correspond to identifiable structures in the network, connecting this work to mechanistic interpretability.

This paper contributes both a theory and a methodology. Key search offers a framework in which training dynamics become predictable: per-band residuals decay as Weibulls, bands activate coarse-before-fine, and the full loss also has a Weibull shape. The methodology — decomposing loss into orthogonal resolution bands and reparameterizing time as cumulative parameter distance — provides a richer set of observables than scalar loss alone, against which this and other theories of learning can be formulated and tested.

Acknowledgments

I would like to thank Luc Baracat and Marija Beljan for helpful discussions.

References

Yasaman Bahri, Ethan Dyer, Jared Kaplan, Jaehoon Lee, and Utkarsh Sharma. Explaining neural scaling laws. *Proceedings of the National Academy of Sciences*, 121(27):e2311878121, 2024. doi: 10.1073/pnas.2311878121.

Charles H. Bennett. Logical depth and physical complexity. In Rolf Herken, editor, *The Universal Turing Machine: A Half-Century Survey*, pages 227–257. Oxford University Press, 1988.

Guy Gur-Ari, Daniel A. Roberts, and Ethan Dyer. Gradient descent happens in a tiny subspace. *arXiv preprint arXiv:1812.04754*, 2018.

Lê Nguyên Hoang and Rachid Guerraoui. Deep learning works in practice. But Does it work in theory? *arXiv preprint arXiv:1801.10437*, 2018.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred A. Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5301–5310. PMLR, 2019.

Utkarsh Sharma and Jared Kaplan. Scaling laws from the data manifold dimension. *Journal of Machine Learning Research*, 23(9):1–34, 2022.

Zhi-Qin John Xu, Yaoyu Zhang, Tao Luo, Yanyang Xiao, and Zheng Ma. Frequency principle: Fourier analysis sheds light on deep neural networks. *Communications in Computational Physics*, 28(5):1746–1767, 2020.

A Experimental Details

A.1 Data and Preprocessing

We use CelebA from HuggingFace (`nielsr/CelebA-faces`). Images are center-cropped to 128×128, then bilinearly resized to 64×64 (align_corners=False), and converted to tensors in $[0, 1]$ via `ToTensor()`. We apply no mean/std normalization and no data augmentation. From the single HuggingFace train split, we use the first 182,599 images as our training set and the last 20,000 as the test set; no separate validation set is held out.

A.2 Architectures

CNN autoencoder (1,829,379 parameters). Encoder: four strided convolutions ($3 \to 32 \to 64 \to 128 \to 256$ channels), each followed by ReLU; flatten to a $256 \times 4 \times 4$ feature map; linear projection to a 128-dimensional latent. Decoder: linear projection back to $256 \times 4 \times 4$, followed by four transpose-convolutional layers ($256 \to 128 \to 64 \to 32 \to 3$), each followed by ReLU except the final layer which uses Sigmoid.

ViT autoencoder (2,779,651 parameters). Encoder: patch size 4 (yielding $16 \times 16 = 256$ patches), embedding dimension 192, 4 transformer blocks, 6 attention heads, MLP ratio 4, GELU activation, learned positional embeddings. Mean pooling followed by a linear projection to a 128-dimensional latent. Decoder is the same transpose-convolutional stack as the CNN.

Both models are trained as plain autoencoders with pixel MSE reconstruction loss.

A.3 Training Configuration

Optimizer: plain Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$) with learning rate 3×10^{-4} and no learning-rate schedule, warmup, weight decay, gradient clipping, or mixed precision. Training runs for 50 epochs at batch size 128, in single-precision `float32` on CUDA.

A.4 Runs and Seeds

We train 12 runs in total: CNN and ViT architectures with seeds 45, 46, 47, 48, 49, 50 (12 runs = 6 CNN + 6 ViT). Each seed determines parameter initialization, data ordering, and test-panel sampling.

A.5 Checkpoint Schedule

Checkpoints are saved at 53 points per run, denser at the beginning to capture early training dynamics. The exact schedule is:

- Epoch 0 (initialization).
- Within epoch 1: epoch 1/32, 1/16, 1/8, 3/16, 1/4, 3/8, 1/2, 5/8, 3/4, 7/8, 1.
- Epochs 1.5 to 10: every 0.5 epoch.
- Epochs 11 to 20: every epoch.
- Epochs 22 to 42: every 2 epochs.
- Epochs 45 and 50.

A.6 Haar Decomposition

The Haar decomposition is computed offline from saved test-panel reconstructions using a standard orthogonal 2D Haar pyramid. Band 1 corresponds to the final LL coefficient (coarsest), and bands 2, 4, 8, 16, 32, 64 correspond to increasingly fine detail subbands. Because the decomposition is orthogonal, band losses sum exactly to total image MSE.

The Haar bands form an orthogonal decomposition of the image error. Let the reconstruction error be

$$e = x - f(x; \theta_t).$$

Haar splits this error into separate resolution bands:

$$e = e_1 + e_2 + e_4 + e_8 + \cdots,$$

where each e_i is the part of the error in band i.

The bands are orthogonal, meaning that different bands have zero inner product:

$$\langle e_i, e_{i'} \rangle = 0 \qquad \text{whenever } i \neq i'.$$

In other words, the error in band i does not overlap with the error in band i' under the pixel inner product. Pixel MSE is the average squared error:

$$\mathrm{MSE}(x, f(x;\theta_t)) = \frac{1}{N}\sum_p e_p^2 = \frac{1}{N}\langle e, e\rangle.$$

Therefore, when we expand the squared error of the full image,

$$\langle e, e\rangle = \left\langle \sum_i e_i, \sum_{i'} e_{i'} \right\rangle = \sum_i \langle e_i, e_i\rangle + \sum_{i\neq i'} \langle e_i, e_{i'}\rangle.$$

The second term vanishes because the bands are orthogonal, so only the within-band terms remain:

$$\langle e, e\rangle = \sum_i \langle e_i, e_i\rangle.$$

Dividing by N, the total image MSE becomes the sum of the band MSEs:

$$\mathrm{MSE}(x, f(x;\theta_t)) = \sum_i L_i(x, f(x;\theta_t)).$$

This equality holds for each individual image and reconstruction pair, not only on average over a dataset. It relies on two facts: the bands must be orthogonal, and the loss must be squared error. If the bands are not orthogonal, cross-terms like $\langle e_1, e_2\rangle$ may be nonzero, so the band losses no longer add exactly to the total loss. If the loss is not squared error, such as L1 or cross-entropy, there is no Pythagorean theorem that gives the same exact additive decomposition.

A.7 Optimizer-Path Measurements

Parameter time $s(t)$ is computed between saved checkpoints:

$$s(t_k) = \sum_{m=1}^{k} \|\theta_m - \theta_{m-1}\|_2,$$

where the sum is taken over all trainable parameters (no exclusions, no per-layer normalization). Because s is accumulated between saved checkpoints rather than per optimizer step, it approximates the true optimizer path by chord lengths between checkpoints. This slightly underestimates the true arc length but preserves the relative ordering of checkpoints, which is what the analyses in this paper require.

Empirically, parameter time is well fit as a power of raw time, $s(t) \approx a + b \cdot t^{\gamma}$. Using a three-parameter nonlinear fit over the full 50-epoch window, the median exponent is $\gamma = 0.368$, with SD $= 0.008$ across runs and median $R^2 = 0.999$. The exponent is different across architectures and window sizes. So γ is best read as a window-specific parameter rather than a universal constant.

Window	Architecture	Runs	Median γ	Median R^2
0–10	CNN	6	0.354	0.999
0–25	CNN	6	0.371	1.000
0–50	CNN	6	0.370	1.000
0–10	ViT	6	0.472	0.999
0–25	ViT	6	0.412	0.999
0–50	ViT	6	0.365	0.999

A.8 Evaluation Panel

Each run has a fixed evaluation panel of 256 images, sampled without replacement from the 20,000 images test split using RNG seed (seed + 3). The panel is fixed across all checkpoints within a run but not shared across runs. Panel reconstructions are stored as float16 NumPy archives.

A.9 Floors

The band floor L_i^∞ is the mean band loss over the last five saved checkpoints. The full-loss floor is the sum of the band floors, $L_{\text{tot}}^\infty = \sum_i L_i^\infty$, so that the additivity $R_{\text{tot}}(s_k) = \sum_i R_i(s_k)$ holds exactly. For comparison, the late-checkpoint average differs from the final-checkpoint floor by at most 0.51% of each band's full dynamic range across all run–band pairs, so the two conventions are effectively interchangeable for this dataset.

A.10 Threshold Crossings

The α-crossing $c_i(\alpha)$ (defined in Section 5.1) is estimated by linear interpolation: if $q_i(s_{k-1}) > \alpha$ and $q_i(s_k) \le \alpha$, then

$$c_i(\alpha) = s_{k-1} + \frac{q_i(s_{k-1}) - \alpha}{q_i(s_{k-1}) - q_i(s_k)} \, (s_k - s_{k-1}) \, .$$

The interpolation is used only to estimate crossing times. For per-band fits, the window starts at the first saved checkpoint after $c_i(0.95)$ and ends at the last saved checkpoint before $c_i(0.01)$. No interpolated residual or loss values enter the least-squares objective. Direct global fits to R_{tot} use the full saved-checkpoint range with no threshold trimming.

A.11 BIC Calculation

BIC is computed from the same checkpoint values and the same linear-scale least-squares objective used for fitting. For fitted values $\widehat{R}(s_k)$ on n saved checkpoints, let

$$\text{SSE} = \sum_{k=1}^{n} \Big(R(s_k) - \widehat{R}(s_k) \Big)^2 .$$

The full BIC formula is

$$\text{BIC} = n \log\left(\frac{\text{SSE}}{n} \right) + p \log n + n \log(2\pi) + n.$$

The last two terms cancel in

$$\Delta\text{BIC} = \text{BIC}_{\text{Weibull}} - \text{BIC}_{\text{exp}},$$

so we compute BIC in a reduced form

$$\text{BIC} = n \log\left(\frac{\text{SSE}}{n} \right) + p \log n.$$

The Weibull fit has $p = 3$ parameters (A, ω, β); the exponential baseline fixes $\beta = 1$ and has $p = 2$ parameters (A, ω).

A.12 Software and AI Assistance

PyTorch 2.7.1, HuggingFace Datasets, NumPy, Pandas, SciPy, PIL. Fitting uses `scipy.optimize.least_squares`. Public code and derived clean artifacts are available at `https://github.com/jurij-jukic/key_search_paper_code`. Heavyweight checkpoints and panel dumps are referenced by manifests and can be materialized separately.

OpenAI's GPT 5.4 and 5.5 were used for orchestrating training runs, and writing the analysis code. Writing this paper was assisted by Anthropic's Claude 4.6, 4.7, and OpenAI's GPT 5.4 and 5.5.

A.13 Hardware and Runtime

Training runs on NVIDIA GPUs (RTX 3090-class; exact model varies by launch manifest). Median per-run wall-clock time: 24 minutes for CNN runs, 89 minutes for ViT runs. Total compute for the 12-run pilot: approximately 12 GPU-hours.

B Weibull Fit Details

B.1 Fit Form and Procedure

We fit the Weibull form

$$R_i(s) = A \cdot \exp\left(-((s - s_{\text{start}})/\omega)^{\beta}\right), \tag{4}$$

where A is the fitted active-window amplitude, not necessarily the initial residual $R_i(s_0)$. We use `scipy.optimize.least_squares` with parameter bounds $A > 0$, $\omega > 0$, $\beta \in [0.05, 8]$. The optimizer is multistarted from several initial parameter guesses to reduce sensitivity to local minima.

For each band, the active window is selected using the crossings defined in Appendix A.10. Within this window, s is rebased so that the first fitted checkpoint has $s = 0$.

B.2 Per-Band Weibull vs. Exponential

For each of 84 run-band fits (12 runs $\times$ 7 bands), we compare Weibull against pure exponential ($\beta = 1$ in the same window) via BIC.

Overall: median $R^2 = 0.997$, median $\Delta\text{BIC} = -19.65$, Weibull preferred in 72/84 fits.

Per-band medians (across 12 runs):

Band	R^2	β	ΔBIC	Wins	ΔBIC ($n/10$)	Wins ($n/10$)
1	0.982	0.90	$+1.14$	4/12	-0.52	12/12
2	0.992	3.13	-19.05	11/12	-1.97	12/12
4	0.995	1.31	-3.58	10/12	-0.49	8/12
8	0.996	1.29	-9.27	12/12	-0.80	11/12
16	0.998	1.28	-50.63	12/12	-4.28	12/12
32	0.999	1.26	-70.92	12/12	-6.03	12/12
64	0.999	1.16	-35.44	11/12	-2.49	11/12

Weibull is decisively preferred over exponential from band 8 upward. Bands 4–64 show β consistently above 1 (initial shoulder and then decay), with $\beta \in [1.16, 1.31]$.

Two coarse bands are less clean: band 1's shape is not distinguishably non-exponential, and band 2 has an inflated $\beta = 3.1$. Both issues likely stem from sparse early checkpoints combined with optimizer instability in the first epoch, when these bands are decaying fastest.

Robustness to effective sample size. Adjacent checkpoints are temporally correlated, so nominal N overestimates effective sample size. Under the conservative adjustment $n_{\text{eff}} = n/10$, bands 16 and 32 remain decisive ($|\Delta\text{BIC}| > 4$); other bands become marginal but Weibull still wins the majority at every band (78/84 overall, median $\Delta\text{BIC} = -1.73$).

B.3 Robustness to Fit-Window Choice

We test whether the conclusions for each band depend on the particular active-window thresholds by repeating the analysis with two narrower windows, 0.90/0.10 and 0.80/0.20, in addition to the primary 0.95/0.01 window.

B.3.1 Per-Band Weibull Fits

Medians across runs under three threshold windows (left edge / right edge). In some cases the medians are applied to less than 12 runs because some runs under narrower windows don't have enough checkpoints to make the data reliable. When there is less than five checkpoints inside the window, the data is omitted.

Band	R^2 (.95/.01)	β (.95/.01)	ΔBIC (.95/.01)	R^2 (.90/.10)	β (.90/.10)	ΔBIC (.90/.10)	R^2 (.80/.20)	β (.80/.20)	ΔBIC (.80/.20)
1	0.982	0.90	$+1.14$	—	—	—	—	—	—
2	0.992	3.13	-19.05	0.979	2.62	-11.88	—	—	—
4	0.995	1.31	-3.58	0.994	1.36	-3.60	0.993	1.60	-5.47
8	0.995	1.29	-9.27	0.993	1.32	-5.05	0.994	1.03	$+1.47$
16	0.998	1.28	-50.63	0.997	1.21	-14.09	0.997	1.10	-8.63
32	0.999	1.26	-70.92	0.999	1.29	-45.73	0.999	1.22	-19.37
64	0.999	1.16	-35.44	0.999	1.17	-36.09	1.000	1.19	-38.79

Fit quality (R^2) is stable across thresholds. ΔBIC magnitudes change but Weibull preference remains strong for bands 16–64; band 8 becomes slightly exponential-favored under the narrowest 0.80/0.20 window.

B.3.2 Band Ordering

Within-run Spearman correlations across 12 runs.

Metric	Window	Median ρ	IQR	Min–max
Onset τ_i	0.95/0.01	0.786	[0.750, 0.786]	[0.750, 0.893]
Onset τ_i	0.90/0.10	0.786	[0.750, 0.786]	[0.750, 0.893]
Onset τ_i	0.80/0.20	0.804	[0.750, 0.929]	[0.750, 0.964]
Width w_i	0.95/0.01	0.964	[0.964, 0.964]	[0.929, 1.000]
Width w_i	0.90/0.10	1.000	[0.964, 1.000]	[0.964, 1.000]
Width w_i	0.80/0.20	1.000	[1.000, 1.000]	[0.964, 1.000]
Floor L_i^∞	0.95/0.01	0.964	[0.964, 0.964]	[0.964, 0.964]
Floor L_i^∞	0.90/0.10	0.964	[0.964, 0.964]	[0.964, 0.964]
Floor L_i^∞	0.80/0.20	0.964	[0.964, 0.964]	[0.964, 0.964]
Initial residual $R_i(s_0)$	0.95/0.01	-0.893	$[-0.893, -0.875]$	$[-0.893, -0.821]$
Initial residual $R_i(s_0)$	0.90/0.10	-0.893	$[-0.893, -0.875]$	$[-0.893, -0.821]$
Initial residual $R_i(s_0)$	0.80/0.20	-0.893	$[-0.893, -0.875]$	$[-0.893, -0.821]$

Onset, width, floor, and initial-residual correlations are essentially invariant to threshold choice.

C Notation

Symbol	Meaning
Data and model	
x	Input image
$f(x;\theta_t)$	Model reconstruction of x at training time t
θ_t	Model parameters at training time t
e	Reconstruction error, $e = x - f(x;\theta_t)$
N	Number of pixels per image
Time	
t	Raw training time (step count)
$s(t)$	Parameter time: cumulative Euclidean distance by raw step t
k	Checkpoint index
s_k	Parameter time at saved checkpoint k
s_0	Parameter time at the initial checkpoint
Resolution bands	
i	Band index; bands correspond to spatial scales $r \in \{1, 2, 4, 8, 16, 32, 64\}$
e_i	Reconstruction error in band i
Loss and residuals	
$L_i(s)$	Band loss: MSE restricted to band i at time s
L_i^∞	Band floor: mean loss over the last five saved checkpoints in band i
$L_{\text{tot}}(s)$	Total loss at time s: $\sum_i L_i(s)$
L_{tot}^∞	Total floor: $\sum_i L_i^\infty$
$R_i(s)$	Band residual: $L_i(s) - L_i^\infty$
$R_i(s_0)$	Initial residual in band i, used for band ordering

Symbol	Meaning
$R_{\mathrm{tot}}(s)$	Total residual: $\sum_i R_i(s) = L_{\mathrm{tot}}(s) - L_{\mathrm{tot}}^{\infty}$
Weibull fits and comparisons	
A	Fitted active-window amplitude in Eq. (4)
ω	Weibull scale parameter
β	Weibull shape parameter
BIC	Bayesian Information Criterion
$\Delta\mathrm{BIC}$	$\mathrm{BIC}_{\mathrm{Weibull}} - \mathrm{BIC}_{\mathrm{exp}}$; negative favors Weibull
$\widehat{R}_i$	Weibull fit for band i
$\widehat{R}_\Sigma$	Band fit reconstruction: $\sum_i \widehat{R}_i$
Threshold crossings	
$q_i(s_k)$	Normalized residual: $R_i(s_k)/R_i(s_0)$
α	Threshold level
$c_i(\alpha)$	α-crossing: interpolated parameter time at which q_i first falls below α
τ_i	Onset: $c_i(0.95)$
w_i	Active width: $c_i(0.01) - c_i(0.95)$
Weibull tail (theory)	
$T_i(x)$	Tail function for band i
A_i^{tail}	Theoretical tail amplitude in Eq. (3)
$x_i(t)$	Frontier position (latent) for band i
ℓ	Key length in the key-search model