Feature Learning and Signal Propagation in Deep Neural Networks

Yizhang Lou [1] **Chris Mingard** [2 3] **Soufiane Hayou** [4]

Abstract

Recent work by Baratin et al. (2021) sheds light on an intriguing pattern that occurs during the training of deep neural networks: some layers *align* much more with data compared to other layers (where the *alignment* is defined as the euclidean product of the tangent features matrix and the data labels matrix). The curve of the alignment as a function of layer index (generally) exhibits an ascent-descent pattern where the maximum is reached for some hidden layer. In this work, we provide the first explanation for this phenomenon. We introduce the *Equilibrium Hypothesis* which connects this alignment pattern to signal propagation in deep neural networks. Our experiments demonstrate an excellent match with the theoretical predictions.

1. Introduction

The empirical success of modern Deep Neural Networks (DNNs) has sparked a growing interest in the theoretical understanding of these models. An important development in this direction was the introduction of the Neural Tangent Kernel (NTK) framework by Jacot et al. (2018), which provides a dual view of Gradient Descent (GD) in function space. The NTK is the dot product kernel of Tangent Features (gradient features), given by

$$K(x, x') = \nabla_\theta f(x) \nabla_\theta f(x')^T,$$

where f is the network output and θ is the vector of model parameters. The NTK has been the subject of an extensive body of literature, both in the NTK regime where the NTK remains constant during training, e.g. (Jacot et al., 2018; Jacot et al., 2020; Hayou et al., 2020; Ghorbani et al., 2019; Yang, 2020, 2019a), and when the NTK changes during training, e.g (Baratin et al., 2021; Yang and Hu, 2021). The latter is called the *feature learning* regime since the tangent features $\nabla_\theta f(x)^T$ evolve during training in some data-dependent directions; we say that tangent features and the NTK *adapt* to the data. A simple way to quantify *how much these features adapt to data* is by measuring the *alignment* between the tangent features and data labels (Baratin et al., 2021), which is given by the normalized euclidean product between the tangent kernel matrix $\hat{K}$ and data labels matrix YY^T (see Section 2).

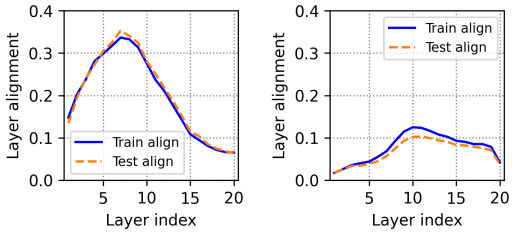


Figure 1: Example Alignment Hierarchies for 20 layer FFNNs trained on Fashion MNIST (L) and CIFAR10 (R).

Role of hidden layers. In the context of DNNs, hidden layers act as successive embeddings that evolve in data-dependent directions during training. However, it is still unclear how different layers contribute to model performance. A possible approach in this direction is the study of feature learning in each layer, i.e. how features adapt to data as we train the model. A simple way to do this is by measuring the change in the values of pre-activations as we train the network; this was used by Yang and Hu (2021) to engineer a network parameterization that maximizes feature learning. To capture the dynamics of GD, Baratin et al. (2021) studied the change in tangent features instead of pre-activations. For each layer l, they measured the alignment between the layer tangent kernel matrix $\hat{K}_l$ (where $K_l(x, x') = \nabla_{\theta_l} f(x) \nabla_{\theta_l} f(x')^T$, θ_l being the vector of parameters in the l^{th} layer). The kernel $\hat{K}_l$ can be seen as the NTK of the network if only the parameters of the l^{th} layer are allowed to change (other layers are frozen, see Section 2.1 for a detailed discussion). The authors demonstrated the existence of an alignment pattern in which the tangent features of some hidden layers are significantly more aligned with data labels compared to other layers. We call this pattern the *Alignment Hierarchy* (illustrated in Fig. 1).

[1]St John's College, University of Oxford, Oxford, UK [2]PTCL, University of Oxford, Oxford, UK [3]Department of Physics, University of Oxford, UK [4]Department of Mathematics, National University of Singapore. Correspondence to: Yizhang Lou <yizhang.lou@sjc.ox.ac.uk>, Soufiane Hayou <hayou@nus.edu.sg>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

In general, the alignment reaches its maximum for some hidden layer and tends to be minimal in the external layers (first and last). To the best of our knowledge, no explanation has been provided for this phenomenon in the literature. In this paper, we propose an explanation based on the theory of signal propagation in randomly initialized neural networks. Our intuition is based on the observation that tangent kernels can be decomposed as an Hadamard product of two quantities linked to how signal propagates in DNNs.

Forward-Backward Decomposition. We show in Section 3 that for a depth L neural network, with proper normalization, the l^{th} layer tangent kernel matrix can be written as

$$\hat{K}_l = \overrightarrow{K}_l \circ \overleftarrow{K}_l \tag{1}$$

where $\overrightarrow{K}_l$ is a kernel that depends only on the l first layers, and $\overleftarrow{K}_l$ is a kernel that is essentially governed by the last $L-l+1$ layers. The $\circ$ denotes the Hadamard product. Intuitively, this decomposition suggests that tangent features are the result of the collaboration between the *forward* kernel $\overrightarrow{K}_l$ and the *backward* kernel $\overleftarrow{K}_l$. For DNNs, it is expected that depth has some layer-dependent effect on the kernels $\overrightarrow{K}_l$ and $\overleftarrow{K}_l$ and therefore on K_l. Understanding the effect of depth on how information propagates has been the subject of a large body of work which constitute what is known as the theory of signal propagation in DNNs. We briefly introduce this theory in the next paragraph (a more detailed discussion is provided in Section 3).

Signal propagation in DNNs. A recent line of research (e.g Hayou et al. (2020), Yang (2020), Yang and Hu (2021), Poole et al. (2016), S. Schoenholz et al. (2017), Lee, Bahri, et al. (2018), and Hayou et al. (2019)) studied the dynamics of signal propagation in randomly initialized DNNs. Under mild conditions, in the infinite (layer) width limit, each neuron $y(.)$ in the network converges in distribution to a Gaussian process (GP) (Neal, 1995; A. Matthews et al., 2018). Hence, in this limit, the covariance kernel captures all the properties of the neurons at initialization (Lee, Bahri, et al., 2018; Neal, 1995)[1]. From a geometric perspective, the correlation/covariance measures the angular distortion between vectors. Hence, the covariance represents a *geometric information*[2]. As shown in Section 3, the kernels $\overrightarrow{K}_l$ and $\overleftarrow{K}_l$ are covariance kernels; $\overrightarrow{K}_l$, resp. $\overleftarrow{K}_l$, represents a notion of forward, resp. backward, geometric information (covariance). Inspired by this observation, we provide an explanation of the feature alignment pattern based on how the geometric information, encoded in $\overrightarrow{K}_l$ and $\overleftarrow{K}_l$, changes with depth. Notably, we prove that these geometric information become degenerate in the limit of large depth which translates to the information being lost as we increase depth. We say that there is a *information loss* as we increase depth and we characterize this loss in the case of fully-connected DNNs. Could this information loss be the reason behind the observed alignment pattern? More precisely, could some notion of balance between information loss in kernels $\overrightarrow{K}_l$ and $\overleftarrow{K}_l$ explain the alignment pattern? We formulate this intuition as the *Equilibrium hypothesis* (EH) which we introduce in Section 3.

[1] GPs are fully characterized by their covariance kernel.

[2] Here, the information refers purely to the covariance between two vectors, and is different from the information-theoretic definition of information.

Our contributions are three-fold. Firstly, we introduce and empirically validate the Equilibrium Hypothesis, which provides an explanation for the alignment pattern. More precisely, we give an explanation for the fact that the alignment peaks at some hidden layer. Secondly, we provide a comprehensive analysis of this hypothesis in the case of fully-connected neural networks. Most notably, we prove that layers with indices $\mathbf{l} = \mathbf{\Theta}(\mathbf{L^{3/5}})$ achieve a notion of *equilibrium* in geometric information in the limit of large depth L. Our experiments yield excellent match between theoretical and empirical results. Finally, we provide an empirical analysis of the connection between the Alignment Hierarchy (illustrated in Fig. 1) and the generalization error.

2. Feature Learning in DNNs

Consider a neural network model consisting of L layers of widths $(N_l)_{1\leq l\leq L}$, $N_0 = d$, and let $\theta = (\theta_l)_{1\leq l\leq L}$ be the flattened vector of weights indexed by the layer's index, and P be the dimension of θ. Given an input $x \in \mathbb{R}^d$, the network is described by the set of equations

$$z_l(x) = \mathcal{F}_l(\theta_l, z_{l-1}(x)), \quad 1 \leq l \leq L,$$

where $\mathcal{F}_l$ is a mapping that defines the l^{th} layer, e.g. fully-connected, convolutional, etc.

The network output function f is given by $f_\theta(x) = \nu(z_L(x)) \in \mathbb{R}^o$ where $\nu : \mathbb{R}^{N_L} \to \mathbb{R}^o$ is a mapping of choice, and o is the dimension of the output, e.g. the number of classes for a classification problem.

We consider a loss function $\mathbb{L} : \mathbb{R}^o \times \mathbb{R}^o \to \mathbb{R}$ and a dataset $\mathcal{D} = \{(x_1, y_1), \dots, (x_n, y_n)\}$. The network is then trained by minimizing the empirical loss $\mathcal{L} : \mathbb{R}^P \to \mathbb{R}$ given by

$$\mathcal{L}(\theta) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{L}(f_\theta(x_i), y_i).$$

Tangent Features. Jacot et al., 2018 introduced the Neural Tangent Kernel (NTK), which provides a dual view of the training procedure; it links gradient updates in parameter space to a kernel gradient descent in function space. The

NTK is given by

$$K_\theta^L(x,x') = \nabla_\theta f_\theta(x)\nabla_\theta f_\theta(x')^T = \sum_{l=1}^{L} \nabla_{\theta_l} f_\theta(x)\nabla_{\theta_l} f_\theta(x')^T \in \mathbb{R}^{o\times o}. \quad (2)$$

The tangent features are the feature maps of the NTK, given by the output gradients w.r.t the network parameters, namely

$$\Psi_\theta(x) := \nabla_\theta f_\theta(x)^T \in \mathbb{R}^{P\times o}. \quad (3)$$

For the sake of simplicity, we remove θ and L in the kernel notation and define $\mathbf{\Psi} \in \mathbb{R}^{P\times on}$, the tangent feature matrix over the training dataset $\mathcal{D}$. $\mathbf{\Psi}$ is the horizontal concatenation of $\Psi(x_1), \ldots, \Psi(x_n)$. The corresponding tangent kernel matrix is given by $\hat{\mathbf{K}} = \mathbf{\Psi}^T\mathbf{\Psi} \in \mathbb{R}^{on\times on}$.

2.1. Quantifying the role of each layer

Lee, S. S. Schoenholz, et al. (2020), Lee, Xiao, et al. (2019), Valle-Perez et al. (2018), and Mingard et al. (2021) demonstrated that neural network based kernel methods (e.g. infinite width NTK regime) can achieve near parity with finite width networks, and have near identical posterior distributions, over a range of architectures (e.g. LSTMs, WideResNet) and datasets (e.g. Cifar10). However, optimizer hyperparameters are known to affect generalization, suggesting an extra layer of complexity (Mingard et al., 2021; Bernstein and Yue, 2021). Furthermore, Hayou et al. (2020) proved that the large depth limit of the NTK regime is trivial in the sense that the limiting NTK has rank 1. This suggests that this kernel regime, where tangent features are fixed at initialization, cannot explain the inductive bias of ultra deep neural networks, and that feature learning (tangent features evolve during training) could be the backbone of generalization in very deep networks. Finite width CNNs operating in the feature learning regime have also been shown to generalise better than their infinite width counterparts (Lee, S. S. Schoenholz, et al., 2020). A question that arises is that of the role of each layer in feature learning – unfortunately there is no consensus on how feature learning should best be measured. One way to approach this question is by analyzing the behaviour of a network where only the parameters in a given layer are allowed to change with gradient updates. We call this approach 'parameter freezing'.

Parameter freezing. The derivations of the results in this paragraph are provided in Appendix C. We omitted the details in the main text to meet space constraints.

Consider a classification task with $o = k$ classes and assume that the dataset is balanced, i.e. $\frac{1}{n}\sum_{i=1}^n y_i \approx \frac{1}{k}\mathbf{1}$, where $\mathbf{1} \in \mathbb{R}^k$ is the vector of ones.

Intuition: One way to measure feature learning in the l^{th} layer is by freezing the parameters in the other $L-1$ layers and tracking the change in network output when we update the parameters of the l^{th} layer.

Given a layer index l, suppose that we freeze all the weights in the other $L-1$ layers and allow the parameters in the l^{th} layer to be updated with a gradient step. Consider the vector $f_t(X) \in \mathbb{R}^{kn}$ which consists of the concatenation of the sequence $(f_t(x_i))_{1\le i\le n}$ (here X refers to the concatenation of $x_1, x_2, \ldots, x_n$). Then, with one gradient step, the update $\delta f_t(X)$ is given by

$$\delta f_t(X) = -\eta\hat{\mathbf{K}}_l(Z_t - Y),$$

where $\hat{\mathbf{K}}_l$ is the tangent kernel matrix for layer l, $Z_t := (\text{softmax}(f_t(x_i)))_{1\le i\le n}, Y \in \mathbb{R}^{on}$, and η is the normalized learning rate (i.e. $\eta = \text{LR}/n$). Hence, the kernel matrix $\hat{\mathbf{K}}_l$ controls the change in the vector $f_t(X)$.

To understand the interaction between $\hat{\mathbf{K}}_l$ and Y, let us see what happens at the first step of gradient descent. At initialization, the output function f is random and has an average accuracy of an random classifier, i.e. a random guess with uniform probability $1/k$ for each class. In this case, the average update is given by

$$\delta f_t(X) \approx -\eta\hat{\mathbf{K}}_l\left(\frac{1}{k}\mathbf{1} - Y\right) \approx \eta\hat{\mathbf{K}}_l\tilde{Y}, \quad (4)$$

where $\tilde{Y} = \left(I - \frac{1}{kn}\mathbf{1}\mathbf{1}^T\right)Y$. $\tilde{Y}$ is a centered version of Y. Using Eq. (4), we obtain

$$\|\delta f_t(X)\| \le \eta\text{Tr}(\hat{\mathbf{K}}_l)\|\tilde{Y}\|,$$

with equality if and only if $\hat{\mathbf{K}}_l$ and $\tilde{Y}$ are aligned, i.e. $\hat{\mathbf{K}}_l \propto \tilde{Y}\tilde{Y}^T$. Hence, the maximum update of the network output is induced by a perfect alignment between $\hat{\mathbf{K}}_l$ and $\tilde{Y}\tilde{Y}^T$.

Conclusion: Assume that only parameters in the l^{th} layer are updated with gradient descent. Then, the alignment between the tangent kernel matrix $\hat{\mathbf{K}}_l$ and the centered data labels matrix $\tilde{Y}\tilde{Y}^T$ controls the magnitude of change in $f_t(X)$ at early training.

Although this analysis is performed at early training, we hypothesize that this alignment quantifies feature learning in each layer during training[3]. Hence, we propose to use this alignment to quantify the role of each layer. This alignment can also be interpreted as a measure of how informative each layer's gradient update is. In fact, if we update all the layers (no parameter freezing), the change in function update projected on data labels vector is approximately

$$\langle\tilde{Y}, \delta f_t(X)\rangle \approx \eta\tilde{Y}^T\hat{\mathbf{K}}\tilde{Y} = \sum_l \eta\text{Tr}(\hat{\mathbf{K}}_l\tilde{Y}\tilde{Y}^T), \quad (5)$$

Hence, $\text{Tr}(\hat{\mathbf{K}}_l\tilde{Y}\tilde{Y}^T)$ quantifies how gradient updates in each layer contribute to output function moving in the direction of the training target.

[3] Fig. 2 shows that the increase in alignments occurs mostly during the first few epochs.

2.2. Centered Kernel Alignment (CKA)

From the analysis in Section 2.1, we define the *centered kernel alignment* between two kernel matrices $\mathbf{K}, \mathbf{K}' \in \mathbb{R}^{on \times on}$ by

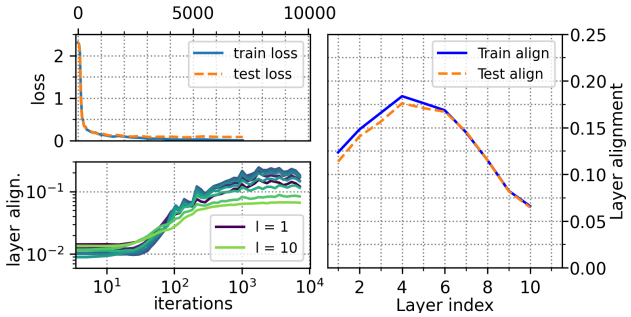
$$A(\mathbf{K}, \mathbf{K}') = \frac{\mathrm{Tr}\left[\mathbf{K}_c \mathbf{K}'_c\right]}{\|\mathbf{K}_c\|_F \|\mathbf{K}'_c\|_F} \tag{6}$$

where $\mathbf{K}_c = \mathbf{CKC}$, $\mathbf{C} = \mathbf{I} - \frac{1}{on}\mathbf{1}\mathbf{1}^T$ is the centering matrix ($\mathbf{1}$ is a vector with all entries being 1), and $\|.\|_F$ is the Frobenius norm. The CKA was used by Baratin et al., 2021 as a measure of feature learning.

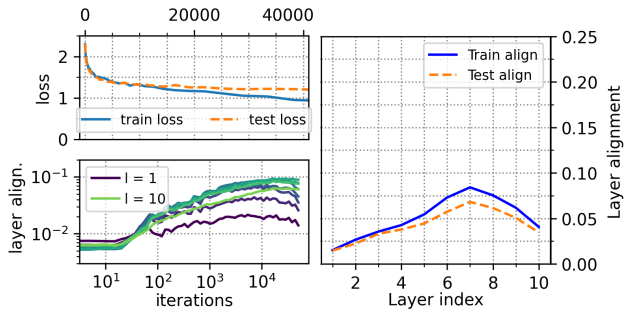
Remark 1. For all kernels $\mathbf{K}, \mathbf{K}'$, we have $A(\mathbf{K}, \mathbf{K}') \in [0, 1]$ with $A(\mathbf{K}, \mathbf{K}') = 1$ if and only if the kernel matrices are colinear.

To quantify the role of the l^{th} layer ($1 \leq l \leq L$), we use $\mathbf{K} = \hat{\mathbf{K}}_l = \mathbf{\Psi}_l^T \mathbf{\Psi}_l$ where $\mathbf{\Psi}_l \in \mathbb{R}^{P_l \times on}$ are the tangent features of layer l (the horizontal concatenation of the tangent features $(\Psi_l(x_i) = \nabla_{\theta_l} f_\theta(x_i))_{1 \leq i \leq n}$), P_l is the dimension of θ_l, and $\mathbf{K}' = YY^T$ where $Y \in \mathbb{R}^{on}$ is the horizontal concatenation of output vectors in the dataset $\mathcal{D}$.

$$A_l := A(\mathbf{K}_l, \mathbf{K}'_l) = \frac{\tilde{Y}^T \mathbf{\Psi}_l^T \mathbf{\Psi}_l \tilde{Y}}{\left\|\mathbf{C}\mathbf{\Psi}_l^T \mathbf{\Psi}_l \mathbf{C}\right\|_F \left\|\tilde{Y}\right\|^2} \tag{7}$$



(a) MNIST



(b) CIFAR10

Figure 2: Layerwise alignment hierarchy for the MNIST and CIFAR10 datasets when trained on an FFNN with depth 10 and width 256. Left hand panels show progression of loss and layer alignment with iterations of SGD. Right hand panel shows layer alignment at the end of training.

2.3. Alignment Hierarchy

(AH) The alignment A_l acts as a measure of how much layers contribute to the performance of the network (Section 2.1). Baratin et al. (2021) observed an interesting hierarchical structure in the alignments A_l for different neural network architectures. During the course of training, the increase in A_l for some middle layers is sharp and significantly larger than the alignments of other layers. We illustrate this pattern in Fig. 2 on MNIST and CIFAR10 datasets with fully-connected networks. It appears that alignments of some layers increase much more effectively with gradient updates over others [4]. We call this pattern the *Alignment Hierarchy*, and aim to understand the reason why the alignment peaks at some hidden layer. For both datasets in Fig. 2, the pattern is similar and shows large alignments for some middle layers. Further empirical results on K-MNIST and FashionMNIST datasets and VGG19/ResNet18 architectures are provided in Appendix E. Motivated by this empirical findings, we formulate the Equilibrium Hypothesis in the next section, where we give an explanation of the Alignment Hierarchy using tools from the theory of signal propagation at initialization.

3. The Equilibrium Hypothesis

In this section, we aim to understand a specific aspect of the AH: *why does the alignment peak in some intermediate layer?* We argue that this is a result of the dynamics of signal propagation in DNNs at initialization. For the sake of simplicity, we restrict our theoretical analysis to fully-connected DNNs, although our results can in principle be extended to other architectures.

Fully-connected Feedforward Neural Network (FFNN). Given an input $x \in \mathbb{R}^d$, and a set of weights $(W_l)_{1 \leq L}$, we consider the following neural network model

$$\begin{aligned} z_1(x) &= W_1 x \\ z_l(x) &= W_l \phi(z_{l-1}(x)), \quad 2 \leq l \leq L, \end{aligned} \tag{8}$$

where $W_l \in \mathbb{R}^{N \times N}$ with $o = N_L = 1$[5], $W_1 \in \mathbb{R}^{N \times d}$, N is the network width, and ϕ is the ReLU activation function given by $\phi(v) = (\max(v_i, 0))_{1 \leq i \leq p}$ for $v \in \mathbb{R}^p$. For each layer, the weights are initialized with i.i.d Gaussian variables $W \sim \mathcal{N}(0, \frac{2}{\text{fan\_in}})$, where 'fan_in' refers to the dimension of the previous layer. This standard initialization scheme is known as the He initialization (He et al., 2015) or the Edge of Chaos initialization (Poole et al., 2016; S. Schoenholz et al., 2017; Hayou et al., 2019).

[4] This also suggests the existence of an implicit layer selection phenomenon during training

[5] For simplicity, we restrict our analysis to rectangular networks with 1D output networks.

3.1. Tangent Kernel decomposition

The tangent kernel at hidden layer l can be expressed as

$$K_l(x,x') = \nabla_{\theta_l} f(x) \cdot \nabla_{\theta_l} f(x') = \sum_{i,j} \phi(z^j_{l-1}(x))\phi(z^j_{l-1}(x')) \frac{\partial f}{\partial z^i_l}(x) \frac{\partial f}{\partial z^i_l}(x').$$

Since K_l is a sum over N^2 terms, we consider the average kernel $\bar{K}_l$ given by $\bar{K}_l = \frac{1}{N^2} K_l$. In matrix form[6], $\bar{K}_l$ can be written as the Hadamard product of two kernels

$$\bar{\mathbf{K}}_l = \overrightarrow{\mathbf{K}}_l \circ \overleftarrow{\mathbf{K}}_l, \tag{9}$$

where $\overrightarrow{K}_l(x,x') = \frac{1}{N}\phi(z_{l-1}(x)) \cdot \phi(z_{l-1}(x'))$ is the *forward* features kernel and $\overleftarrow{K}_l(x,x') = \frac{1}{N} \frac{\partial f_{l:L}}{\partial z}(z_l(x)) \cdot \frac{\partial f_{l:L}}{\partial z}(z_l(x'))$ is the *backward* tangent features kernel, where $f_{l:L}$ is the function that maps the l^{th} layer to the network output. The above decomposition illustrates the *collaborative* roles played by kernels $\overrightarrow{K}$ and $\overleftarrow{K}$ in constructing the tangent features at layer l. To depict the role of each kernel, we use some tools from the theory of signal propagation at initialization.

3.2. Signal propagation at initialization

Consider an FFNN of type (8). The weights W are randomly initialized. Hence, the network neurons and output are random processes at initialization. Understanding the properties of such processes is crucial for both training and generalization (S. Schoenholz et al., 2017; Hayou et al., 2019). It turns out that in the limit of infinite width $N \to \infty$, the neurons act as Gaussian processes. To see this, consider the simple case of a two layers FFNN. Since the weights are i.i.d, neurons $\{z^i_1(.)\}_{i\in[1:N]}$ are also iid Gaussian processes with covariance kernel given by $\mathbb{E}_W[z^i_1(x)z^i_1(x')] = \frac{2x\cdot x'}{d}$. Using the Central Limit Theorem, as $N \to \infty$, $z^i_2(x)$ is a Gaussian variable for any input x and index $i \in [1:o]$. Moreover, the random variables $\{z^i_2(x)\}_{i\in[1:o]}$ are iid. Hence, the processes $z^i_2(.)$ can be seen as independent (across i), centered Gaussian processes with some covariance kernel q_2. This Gaussian process limit of FFNNs was first proposed by Neal (1995) in the single layer case and was extended to multi-layer networks by A. Matthews et al. (2018) where the authors showed that in l^{th} layer, neurons become i.i.d Gaussian processes with covariance kernel q_l in the limit $N \to \infty$. This result holds for all standard neural network architectures (Yang, 2019a). A more complete review of this theory is provided in Appendix A. The covariance kernel $q_l(x,x')$ is a measure of the angular distortion between the vectors $z^i_l(x)$ and $z^i_l(x')$. Thus, the covariance kernel carries some information on how inputs propagate within the network. We formalize this notion of information in the next definition.

Definition 1 (Geometric information)**.** *Given random weights W, we say that a kernel function k is a geometric information if it can be expressed as $k(x,x') = \mathbb{E}_W[g(W,x)g(W,x')]$ for some function $g : \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}$.*

Hereafter, we simply use 'information'[7] to refer to the geometric information in Definition 1. Recall the kernel decomposition given by Eq. (9)

$$\bar{\mathbf{K}}_l = \overrightarrow{\mathbf{K}}_l \circ \overleftarrow{\mathbf{K}}_l.$$

The kernels $\overrightarrow{K}_l$ and $\overleftarrow{K}_l$ depend on random weights W and thus are random. We propose to study the average behaviour instead, where we consider the average kernels. For $\overrightarrow{K}_l$, the average kernel is given $\mathbb{E}_W[\overrightarrow{K}_l(x,x')] = \mathbb{E}_W[\phi(z^1_{l-1}(x))\phi(z^1_{l-1}(x'))]$ (since the neurons z^j_l are identically distributed) which represents a geometric information as per Definition 1. We call this average kernel the forward information. A standard result in signal propagation is that kernels $\overrightarrow{K}_l$ and $\overleftarrow{K}_l$ converge to their corresponding expected value in the limit of infinite width (Yang, 2020; S. Schoenholz et al., 2017; Hayou et al., 2019) which justifies our choice of the average kernel $\mathbb{E}_W[\overrightarrow{K}_l(x,x')]$. A similar result holds for $\overleftarrow{K}_l$. Let us formalize these definitions.

Definition 2. *Given a layer index l, we define the forward information $I^f_{l,N}$ by*

$$I^f_{l,N}(x,x') = \mathbb{E}_W\left[\phi(z^1_{l-1}(x))\phi(z^1_{l-1}(x'))\right],$$

where the expectation is taken w.r.t W. Similarly, the backward information $I^b_{l,N}(x,x')$ is defined by

$$I^b_{l,N}(x,x') = \mathbb{E}_W\left[\frac{\partial f_{l:L}}{\partial z^1}(z_l(x))\frac{\partial f_{l:L}}{\partial z^1}(z_l(x'))\right].$$

$I^f_{l,N}$ and $I^b_{l,N}$ are geometric information in accordance with Definition 1.

3.3. Information loss in the large depth limit

A classical result in the theory of signal propagation is that the information deteriorates with depth L (S. Schoenholz et al., 2017; Hayou et al., 2019) in the sense that the covariance kernels converge to trivial kernels (e.g. constant kernels) in the limit of infinite depth. This is a natural result of the randomness that adds to the neurons with each additional layer. This deterioration occurs with some rate (convergence rate to the trivial kernel w.r.t to L) which we call the information loss in the following definition. For two non-negative sequences $(a_n)_{n\geq 0}, (b_n)_{n\geq 0}$, we write

[6] Bold characters in Eq. (9) refer to kernel matrices and not kernels

[7] This is different from the information-theoretic definition of information.

$a_n = \Theta_n(b_n)$ if there exists two constant $M_1, M_2 > 0$ such that for all n, $M_1\, b_n \leq a_n \leq M_2\, b_n$.

Definition 3 (Information Loss ($\mathcal{IL}$)). *Let $(g_n(.))_{n\geq 0}$ be a sequence of real-valued functions defined on some set $\mathcal{C} \subset \mathbb{R}^m, m \geq 1$. Assume that g_n converges uniformly to some constant κ as $n \to \infty$ and that there exists a non-negative sequence $(r_n)_{n\geq 0}$ such that $\sup_{t\in\mathcal{C}} |g_n(t) - \kappa| = \Theta_n(r_n)$. We say that g_n has an information loss of order r_n.*

The $\mathcal{IL}$ characterizes the rate at which the sequence $(g_n(t))_{n\geq 1}$ 'forgets' the input t since, by definition, the limiting value κ is independent of t[8]. In our case, we would expect similar behaviour of the forward information $I^f_{l,N}$ in the limit $l \to \infty$[9], and the backward information $I^b_{l,N}$ in some $(l, L)-$dependent limit(see Section 3.4). For instance, assume that L is large and consider a small l. Then, the forward information has minimal information loss (forward information loss occurs in the limit $l \to \infty$) while the backward information suffers from deterioration as it depends on the $L - l + 1$ last layers, and thus it suffers from the accumulated randomness as it travels back through the network. The opposite happens when l is large, e.g. $l \sim L$. This antagonistic roles of the forward/backward information is key in understanding the behaviour of the tangent kernel K_l: there exists a layer index l_0 for which the information loss for forward and backward information is comparable. By Eq. (1), we expect K_l to suffer from the deterioration that affects either the forward/backward information. Hence, we hypothesize that a layer with comparable forward/backward information loss is better conditioned to align with data.

The Equilibrium Hypothesis (EH). *Let $\mathcal{IL}^f_{l,N}$, resp. $\mathcal{IL}^b_{l,N}$, be the information loss of the sequence $(I^f_{l,N})_{l\geq 1}$, resp. $(I^b_{l,N})_{l\geq 1}$. The layers with the highest alignments with data labels are the ones that satisfy the equilibrium property*

$$\mathcal{IL}^f_{l,N} = \Theta\left(\mathcal{IL}^b_{l,N}\right).$$

The EH conjectures that balanced information loss between forward and backward information is related to high alignment with data. Our intuition is as follows: balance between forward and backward information at layer l relates to informative updates of layer l's parameter θ_l, which corresponds to informative updates in the tangent feature $\mathbf{\Psi}_l$ leading to greater alignment with data. To see this for an FFNN, consider function update:

$$\begin{aligned}\delta f_t(X) &= -\eta \hat{\mathbf{K}} \nabla_f \mathbb{L}(f(X), Y) \\ &= -\eta \sum_l \hat{\mathbf{K}}_l \nabla_f \mathbb{L}(f(X), Y),\end{aligned}$$

where the contribution of updating θ_l to the overall change in $f_t(X)$ is $-\eta \hat{\mathbf{K}}_l \nabla_f \mathbb{L}(f(X), Y)$, where $\eta = \text{LR}/n$ is the normalized learning rate. Since the kernel matrices satisfy $\bar{\mathbf{K}}_l = \overrightarrow{\mathbf{K}}_l \circ \overleftarrow{\mathbf{K}}_l$, we expect that any deterioration of the kernels $\overrightarrow{K}_l$ and $\overleftarrow{K}_l$ would affect K_l. Our intuition is that the parameter update in layer l benefits from the equilibrium property which guarantees that none of the kernels is more deteriorated than the other. This ensures that the function space update benefits from both forward and backward geometric information. Note that at initialization, with the same analysis of Section 2.1, the function update due to θ_l can be approximated by $\delta f_t(X) \approx \eta \hat{\mathbf{K}}_l \tilde{Y}$. This suggest that the high alignment between tangent features $\mathbf{\Psi}_l(X)$ and data labels Y is associated with informative parameter update. A more in-depth discussion of this result is provided in Appendix B.4. In addition, informative parameter update is associated with informative tangent feature update (Appendix B.4).

3.4. The Equilibrium in infinite width FFNNs

For FFNNs, we provide a comprehensive analysis of the Equilibrium property in the infinite width limit. We characterize the layers where the equilibrium is achieved and we confirm our theoretical findings with empirical results. For the sake of simplicity, we restrict our theoretical analysis to the sphere $\sqrt{d}\,\mathbb{S}^d = \{x \in \mathbb{R}^d, \|x\| = \sqrt{d}\}$ where $\|.\|$ is the euclidean norm. The generalization to $\mathbb{R}^d$ is straightforward. To avoid issues with col-linearity in the dataset, we consider the set E_ϵ, parameterized by $\epsilon \in (0, 1)$, defined by

$$E_\epsilon = \{(x, x') \in (\sqrt{d}\,\mathbb{S}^d)^2 : \frac{1}{d} x \cdot x' < 1 - \epsilon\} \qquad (10)$$

The next result characterizes the information loss $\mathcal{IL}$ of the forward/backward information defined above in the limit of infinite width ($N \to \infty$). In this limit, the forward information has an information loss of l^{-2}.

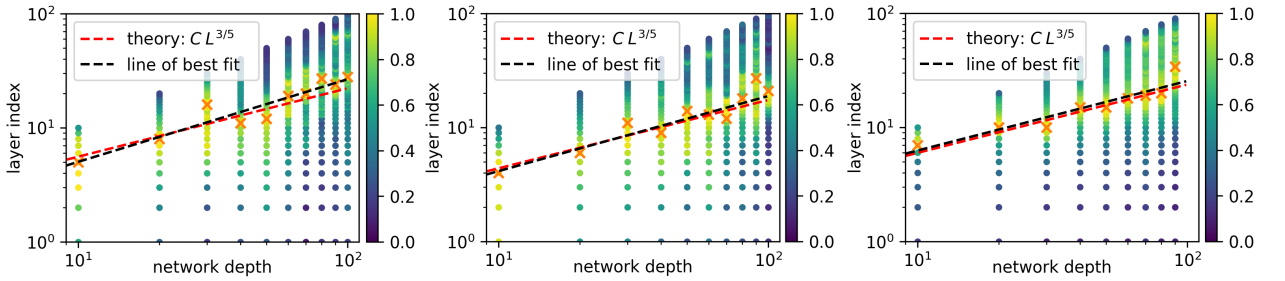
Theorem 1 (Forward $\mathcal{IL}$). *Let $\epsilon \in (0, 1)$, and consider the set E_ϵ as in Eq. (10). Define $I^f_{l,\infty}(x, x') := \lim_{N\to\infty} I^f_{l,N}(x, x')$ for all $x, x' \in \mathbb{R}^d$. We have that*

$$\sup_{(x,x')\in E_\epsilon} |I^f_{l,\infty}(x, x') - 1/2| = \Theta_l(l^{-2}).$$

Theorem 1 is a corollary of a previous result that appeared in Hayou et al. (2020). The proof of the latter relies on an asymptotic analysis of the forward covariance kernel in the limit of large l, coupled with a uniform bounding of the convergence rate (See Appendix B for more details).
The forward information $I^f_{l,N}$ does not depend on depth L. On the other hand, the backward information $I^b_{l,N}$ depends both on l and the depth L. Therefore, in order to study the asymptotic information loss, we should specify how l grows

[8] Note that $\mathcal{IL}$ is unique up to a Θ factor, e.g. an $\mathcal{IL}$ of n^{-1} is the same as an $\mathcal{IL}$ of $n^{-1} \times (2 + n^{-1})$

[9] Note that $I^f_{l,N}$ does not depend on the network depth L.



(a) MNIST (b) FashionMNIST (c) CIFAR10

Figure 3: Data with $x = 10j$ in the plot corresponds to layer alignments for a FFNN with depth $10j$ trained on the MNIST/FashionMNIST/CIFAR10 datasets. The brighter the color, the closer the corresponding layer's alignment is to the maximum alignment across all layers. x indicates the layer where largest alignment occurs. See Fig. 8 for further experiments on Fashion MNIST showing larger learning rates decrease the y-intercept (alignment peaks in earlier layers).

relatively to L. In the next result, we study the two cases where $l \ll L$ or $l = \lfloor \alpha L \rfloor$.

Theorem 2 (Backward $\mathcal{IL}$)**.** *Let $\epsilon \in (0, 1)$, and consider the set E_ϵ as in Eq. (10). Define $I^b_{l,\infty}(x, x') := \lim_{N\to\infty} I^b_{l,N}(x, x')$ for all $x, x' \in \mathbb{R}^d$. Then,*

- *If $l = \lfloor \alpha L \rfloor$ where $\alpha \in (0, 1)$ is a constant, then there exists a constant μ such that in the limit $L \to \infty$,*

$$\sup_{(x,x')\in E_\epsilon} |I^b_{l,\infty}(x, x') - \mu| = \Theta_L(\log(L)L^{-1})$$

- *In the limit $l, L \to \infty$ with $l/L \to 0$,*

$$\sup_{(x,x')\in E_\epsilon} |I^b_{l,\infty}(x, x')| = \Theta_{l,L}((L/l)^{-3}).$$

A key ingredient in the proof of Theorem 2 in the so-called Gradient Independence assumption. In the literature on signal propagation at initialization (e.g. (Poole et al., 2016; S. Schoenholz et al., 2017; Hayou et al., 2019; Yang and S. Schoenholz, 2017; Yang, 2019b; Xiao et al., 2018)), results on gradient backpropagation rely on the assumption that the weights used for backpropagation are independent from the ones used for forward propagation. Yang (2020) showed that this assumption yields exact computations of gradient covariance and NTK in the infinite width limit. We refer the reader to Appendix A.3 for more details.
In the case of infinite width FFNN, using Theorems 1 and 2, we show which layers satisfy the equilibrium property.

Corollary 1 (Equilibrium)**.** *Under the conditions of Theorems 1 and 2, the equilibrium for an FFNN is achieved for layers with index*

$$l = \Theta_L(L^{3/5})$$

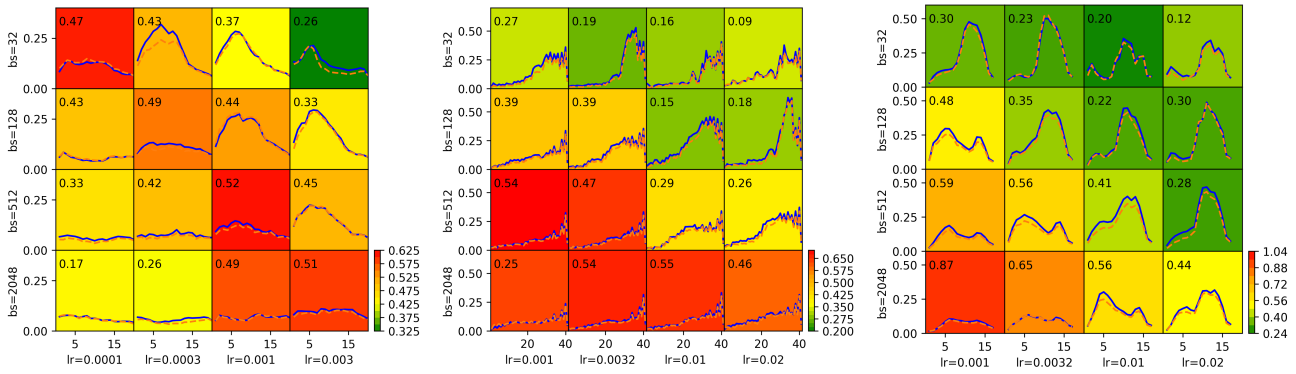
where L is the network depth.

Corollary 1 indicates that layers that satisfy the equilibrium property verify $l = \Theta_L(L^{3/5})$. In logarithmic scale, this implies $\log(l) \in \left[\frac{3}{5}\log(L) + C_1, \frac{3}{5}\log(L) + C_2\right]$ where C_1, C_2 are constants that depends on ϵ from Theorems 1 and 2. Fig. 3 shows the layer alignments A_l's after training an FFNN (width 256) on different datasets. We fit the line $\log(l) = 3/5\log(L) + C$ by finding the constant C that minimizes the squared error. We also perform a simple linear regression to see if the slope is close to $3/5$ (line of best fit). All experiments show an excellent match with the theoretical line $3/5\log L + C$ (which was derived for infinite width networks).

4. Alignment and Hyperparameters

Generalization and feature learning have been linked to optimizer hyperparameter choices (e.g. (Keskar et al., 2016)). While this paper largely focuses on the $L^{3/5}$ scaling law in the EH ($l = \Theta(L^{3/5})$) it would be incomplete without a brief discussion of the constant term in this equation (the y-intercept in Fig. 3). In this section, we observe that the layer index with the greatest alignment can be significantly affected by choice of optimizer hyperparameters, and good generalization is associated with a well defined peak away from the last layer. See Fig. 8 for clear demonstration of the change in C with learning rate.

To understand (1) the effect of optimizer hyperparameters on the alignment and (2) the impact of the Alignment Hierarchy on the generalization error, we trained multiple models with different datasets with a selection of batch sizes, learning rates and optimizers. We show the corresponding AH pattern, generalization error and generalization gap for CIFAR10 on Resnet18 and VGG19 in Fig. 4. The colour corresponds to the final test loss, and the number to the loss gap. Fig. 4 weakly suggests that large alignments with data labels, especially for the middle layers, correlate with good generalization properties. Note that the peak for the FFNN occurs in the early layers; VGG the middle layers, and for Resnet18 very near the last layer. If the EH is also valid for resnets, the peak should move towards the middle layers as depth increases. This is left for future work. For further experiments, including on random labels, see Appendix E.



(a) FFNN on Fashion MNIST (b) Resnet18 on CIFAR10 (c) VGG19 on CIFAR10

Figure 4: Alignment hierarchy as a function of batch size and learning rate for (a) Fashion MNIST on a FFNN (b) CIFAR10 on a Resnet18 and (c) CIFAR10 on a VGG19. The location of the peak is different for each dataset, architecture and hyperparameter combination – note that AH cannot be seen here, as it predicts a scaling law with layers, but not the constants (see Corollary 1). The background colour denotes the test loss, and the number in the top left corner is the generalisation loss gap. Each model was trained stopping after a train loss of 0.1 or 500 epochs, whichever was sooner. There is a clear positive correlation between the height of the peak and the overall generalisation error for each example given here.

Batch size and learning rate dependence Fig. 4 shows that the AH is strongly affected by both batch size and learning rate. Typically, smaller batch sizes and larger learning rates lead to better generalisation where convergence is possible. They also lead to a larger peak in the AH, and the peak shifts towards the center. Although it is expected that learning rate and batch size affect training, we do not currently have a theoretical explanation for these effects on the AH.

Connection to Stable Rank, Alignment & Fisher Information Baratin et al., 2021; Oymak et al., 2019 observed a correlation between improved generalisation and (1) an increasing majority of the singular values of $\mathbf{\Psi}$ are very small with a few very large (2) the label vector Y is increasingly aligned with the large singular directions in $\mathbf{\Psi}$. These are the conditions for maximsing CKA. In Appendix G, we extend these results to Layerwise CKA. The layerwise CKA A_l can be decomposed into a product of the inverse square root of its stable rank and the correlation term between the eigenvectors and the label vector Y. The stable rank measures an effective dimension of the internal representations of the neural network. CIFAR10 with degrees of data randomisation was trained on VGG19, and showed a lower overall layerwise stable rank for the better generalising models. The alignment term varied the most across generalisation errors, with much greater alignment (in the later layers) for the best generalising model than the others. Larger overall alignment therefore correlates with lower stable rank; and an earlier peak appears to coincide with earlier alignment between Y and large singular directions in $\mathbf{\Psi}$. We might expect lower dimensional internal representations and earlier, larger alignment to correlate with good generalisation. We also investigated Fisher Information Matrix $\mathbf{J} = \mathbf{\Psi}\mathbf{\Psi}^T$ (for MSE loss), and $\mathbf{K}$. We use observations from (Maddox et al., 2020) to link properties of $\mathbf{J}$ that coincide with larger CKA to explain why more alignment is likely to correlate with better generalisation.

5. Related work

To the best of our knowledge, the AH has only been discussed in (Baratin et al., 2021) (see also (Nguyen et al., 2020) for work correlating internal layer representations with each other, using a CKA-based method). However, the significance of AH goes beyond a simple feature learning pattern. The AH can be seen a *structural* implicit regularization effect, i.e. a regularization effect that is purely induced by the depth. Traditionally, implicit regularization refers to any hidden regularization effect induced by the training algorithm. For example, it is widely believed that the implicit regularization effect of Stochastic Gradient Descent (SGD) (He et al., 2015; Lecun et al., 1998; Krizhevsky et al., 2012) is mainly driven by the small-batch sampling noise (Jastrzebski et al., 2018). However, recent empirical findings such as (Wu, Zhu, et al., 2017; Geiping et al., 2021) demonstrated that DNNs can still achieve high accuracy on some image datasets with full-batch GD. Goyal et al. (2017) demonstrated that increasing batch size by several orders of magnitude on ImageNet does not affect generalization error significantly, suggesting that classical implicit regularization theories that rely on SGD noise to explain generalization are not sufficient to explain why neural networks generalize *at all* (further backed up by the near parity achieved by kernel methods (Mingard et al., 2021)). These results suggest that implicit regularization might occur via other mechanisms than previously thought, one of them could this purely structural effect that results in the Alignment Hierarchy. Given interest in understanding how the amount of feature learning

affects generalization and transfer learning, we believe this is a promising topic for future work.

6. Conclusion and Limitations

In this paper, we introduced the Equilibrium Hypothesis (EH) which connects information flow at initialization to tangent features alignment with data labels. the EH explains the alignment hierarchy, illustrated in Fig. 2. Our empirical results showed an excellent match with the theoretical prediction $l = \Theta(L^{3/5})$ for FFNN on different datasets in Fig. 3, and presented empirical evidence that earlier alignment correlates with better generalisation Fig. 4. Finally, we used connections between layerwise CKA, the stable rank and Fisher Information to present a theoretical case for this observation. There are still multiple open questions to answer, e.g. the impact of the architecture on the alignment hierarchy. As demonstrated in Fig. 4, it seems that the alignment pattern is sensitive to the choice of the architecture. We leave this topic for future work.

References

Baratin, A., T. George, C. Laurent, R. D. Hjelm, G. Lajoie, P. Vincent, and S. Lacoste-Julien (2021). *Implicit Regularization via Neural Feature Alignment*. arXiv: 2008.00938 [cs.LG].

Jacot, A., F. Gabriel, and C. Hongler (2018). "Neural Tangent Kernel: Convergence and Generalization in Neural Networks". In: *Advances in Neural Information Processing Systems*.

Jacot, A., F. Gabriel, and C. Hongler (2020). *The asymptotic spectrum of the Hessian of DNN throughout training*. arXiv: 1910.02875 [cs.LG].

Hayou, S., A. Doucet, and J. Rousseau (2020). "Mean-field Behaviour of Neural Tangent Kernel for Deep Neural Networks". *arXiv preprint arXiv:1905.13654*.

Ghorbani, B., S. Krishnan, and Y. Xiao (2019). "An investigation into neural net optimization via hessian eigenvalue density". In: *International Conference on Machine Learning*. PMLR, pp. 2232–2241.

Yang, G. (2020). "Tensor Programs III: Neural Matrix Laws". *arXiv preprint arXiv:2009.10685*.

– (2019a). "Tensor programs i: Wide feedforward or recurrent neural networks of any architecture are Gaussian processes". *arXiv preprint arXiv:1910.12478*.

Yang, G. and E. Hu (2021). "Tensor Programs IV: Feature Learning in Infinite-Width Neural Networks". *ICML 2021*.

Poole, B., S. Lahiri, M. Raghu, J. Sohl-Dickstein, and S. Ganguli (2016). "Exponential expressivity in deep neural networks through transient chaos". *30th Conference on Neural Information Processing Systems*.

Schoenholz, S., J. Gilmer, S. Ganguli, and J. Sohl-Dickstein (2017). "Deep Information Propagation". In: *International Conference on Learning Representations*.

Lee, J., Y. Bahri, R. Novak, S. Schoenholz, J. Pennington, and J. Sohl-Dickstein (2018). "Deep Neural Networks as Gaussian Processes". In: *International Conference on Learning Representations*.

Hayou, S., A. Doucet, and J. Rousseau (2019). "On the Impact of the Activation Function on Deep Neural Networks Training". In: *International Conference on Machine Learning*.

Neal, R. (1995). *Bayesian Learning for Neural Networks*. Vol. 118. Springer Science & Business Media.

Matthews, A., J. Hron, M. Rowland, R. Turner, and Z. Ghahramani (2018). "Gaussian Process Behaviour in Wide Deep Neural Networks". In: *International Conference on Learning Representations*.

Lee, J., S. S. Schoenholz, J. Pennington, B. Adlam, L. Xiao, R. Novak, and J. Sohl-Dickstein (2020). "Finite versus infinite neural networks: an empirical study". *arXiv preprint arXiv:2007.15801*.

Lee, J., L. Xiao, S. Schoenholz, Y. Bahri, R. Novak, J. Sohl-Dickstein, and J. Pennington (2019). "Wide neural networks of any depth evolve as linear models under gradient descent". *Advances in neural information processing systems* 32, pp. 8572–8583.

Valle-Perez, G., C. Q. Camargo, and A. A. Louis (2018). "Deep learning generalizes because the parameter-function map is biased towards simple functions". *arXiv preprint arXiv:1805.08522*.

Mingard, C., G. Valle-Pérez, J. Skalse, and A. A. Louis (2021). "Is SGD a Bayesian sampler? Well, almost". *Journal of Machine Learning Research* 22.79, pp. 1–64.

Bernstein, J. and Y. Yue (2021). "On the Implicit Biases of Architecture & Gradient Descent". arXiv: 2110.04274 [cs.LG].

He, K., X. Zhang, S. Ren, and J. Sun (2015). "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification". *ICCV*.

Yang, G. and S. Schoenholz (2017). "Mean field residual networks: On the edge of chaos". In: *Advances in neural information processing systems*, pp. 7103–7114.

Yang, G. (2019b). "Scaling limits of wide neural networks with weight sharing: Gaussian process behavior, gradient independence, and neural tangent kernel derivation". *arXiv preprint arXiv:1902.04760*.

Xiao, L., Y. Bahri, J. Sohl-Dickstein, S. S. Schoenholz, and P. Pennington (2018). "Dynamical Isometry and a Mean Field Theory of CNNs: How to Train 10,000-Layer Vanilla Convolutional Neural Networks". *ICML 2018*.

Keskar, N. S., D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang (2016). "On large-batch training for deep learning: Generalization gap and sharp minima". *arXiv preprint arXiv:1609.04836*.

Oymak, S., Z. Fabian, M. Li, and M. Soltanolkotabi (2019). "Generalization guarantees for neural networks via harnessing the low-rank structure of the jacobian". *arXiv preprint arXiv:1906.05392*.

Maddox, W. J., G. W. Benton, and A. G. Wilson (2020). "Rethinking Parameter Counting in Deep Models: Effective Dimensionality Revisited". *arXiv preprint arXiv:2003.02139*.

Nguyen, T., M. Raghu, and S. Kornblith (2020). "Do wide and deep networks learn the same things? uncovering how neural network representations vary with width and depth". *arXiv preprint arXiv:2010.15327*.

Lecun, Y., L. Bottou, Y. Bengio, and P. Haffner (1998). "Gradient-based learning applied to document recognition". *Proceedings of the IEEE* 86.11, pp. 2278–2324.

Krizhevsky, A., I. Sutskever, and G. E. Hinton (2012). "ImageNet Classification with Deep Convolutional Neural Networks". In:

Jastrzebski, S., Z. Kenton, D. Arpit, N. Ballas, A. Fischer, Y. Bengio, and A. J. Storkey (2018). "Finding flatter minima with sgd". In: *ICLR (Workshop)*.

Wu, L., Z. Zhu, et al. (2017). "Towards understanding generalization of deep learning: Perspective of loss landscapes". *arXiv preprint arXiv:1706.10239*.

Geiping, J., M. Goldblum, P. E. Pope, M. Moeller, and T. Goldstein (2021). *Stochastic Training is Not Necessary for Generalization*. arXiv: 2109.14119 [cs.LG].

Goyal, P., P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He (2017). "Accurate, large minibatch sgd: Training imagenet in 1 hour". *arXiv preprint arXiv:1706.02677*.

Lillicrap, T., D. Cownden, D. Tweed, and C. Akerman (2016). "Random synaptic feedback weights support error backpropagation for deep learning". *Nature Communications* 7.13276.

Karakida, R., S. Akaho, and S.-i. Amari (2019). "Universal statistics of fisher information in deep neural networks: Mean field approach". In: *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, pp. 1032–1041.

Geiger, M., S. Spigler, S. d'Ascoli, L. Sagun, M. Baity-Jesi, G. Biroli, and M. Wyart (2019). "Jamming transition as a paradigm to understand the loss landscape of deep neural networks". *Physical Review E* 100.1, p. 012115.

Shan, H. and B. Bordelon (2021). *Rapid Feature Evolution Accelerates Learning in Neural Networks*. arXiv: 2105.14301 [stat.ML].

Kulkarni, M. and S. Karande (2017). "Layer-wise training of deep networks using kernel similarity". *arXiv preprint arXiv:1703.07115*.

Matthews, A. G. d. G., J. Hron, R. E. Turner, and Z. Ghahramani (2017). "Sample-then-optimize posterior sampling for bayesian linear models". In: *NeurIPS Workshop on Advances in Approximate Bayesian Inference*.

Rudelson, M. and R. Vershynin (2007). "Sampling from large matrices: An approach through geometric functional analysis". *Journal of the ACM (JACM)* 54.4, 21–es.

Roy, O. and M. Vetterli (2007). "The effective rank: A measure of effective dimensionality". In: *2007 15th European signal processing conference*. IEEE, pp. 606–610.

Amari, S.-i. and H. Nagaoka (2000). *Methods of information geometry*. Vol. 191. American Mathematical Soc.

Neyshabur, B., S. Bhojanapalli, and N. Srebro (2017). "A pac-bayesian approach to spectrally-normalized margin bounds for neural networks". *arXiv preprint arXiv:1707.09564*.

A. Review of Signal propagation theory

The signal propagation theory in the context of neural networks deals precisely with the distortion of the information carried by the output as it travels through the network. Most results in this theory (see e.g. Jacot et al. (2018), Jacot et al. (2020), Hayou et al. (2020), Yang (2020), Poole et al. (2016), S. Schoenholz et al. (2017), and Hayou et al. (2019)) consider the infinite width limit as it allows the derivation of closed-form expressions. Infinite width networks are also naturally overparameterized (infinite number of parameters) and therefore might offer some theoretical insights on the overparameterized regime.

Fully-connected FeedForward Neural Networks (FFNN). Given an input $x \in \mathbb{R}^d$, and a set of weights and bias $(W_l, b_l)_{1\leq L}$, the forward propagation is given by

$$z_1(x) = W_1 x + b_1 \tag{1}$$
$$z_l(x) = W_l \phi(z_{l-1}(x)), \quad 2 \leq l \leq L, \tag{2}$$

where $W_l \in \mathbb{R}^{N_l \times N_{l-1}}$, and ϕ is the ReLU activation function given by $\phi(v) = \max(v, 0)$ for $v \in \mathbb{R}$. The dimension of the parameter space is $P = \sum_{l=0}^{L-1} (N_l + 1) N_{l+1}$ where we denote $N_0 := d$. For each layer, the weights are initialized with i.i.d Gaussian variables $W_l^{ij} \sim \mathcal{N}(0, \frac{2}{N_{l-1}})$.

A.1. Forward propagation

When we take the limit $N_{l-1} \to \infty$ recursively over l, this implies, using Central Limit Theorem, that $z_l^i(x)$ is a Gaussian random variable for any input x. The convergence rate to this limiting Gaussian distribution is given $\mathcal{O}(1/\sqrt{N_{l-1}})$ (standard Monte Carlo error). More generally, an approximation of the random process $z_l^i(.)$ by a Gaussian process was first proposed by Neal, 1995 in the single layer case and has been extended to the multiple layer case by Lee, Bahri, et al., 2018 and A. Matthews et al., 2018. The limiting Gaussian process kernels follow a recursive formula given by, for any inputs $x, x' \in \mathbb{R}^d$

$$\begin{aligned}\kappa_l(x, x') &= \mathbb{E}[z_l^i(x) z_l^i(x')] \\ &= 2\,\mathbb{E}[\phi(z_{l-1}^i(x))\phi(z_{l-1}^i(x'))] \\ &= 2\,\Psi_\phi(\kappa_{l-1}(x, x), \kappa_{l-1}(x, x'), \kappa_{l-1}(x', x')),\end{aligned}$$

where Ψ_ϕ is a function that only depends on ϕ. This provides a simple recursive formula for the computation of the kernel κ^l; see, e.g., Lee, Bahri, et al., 2018 for more details.

A.2. Gradient Independence

In the literature of infinite width DNNs, a standard assumption in prior literature is that of the gradient independence which is similar in nature to the concept of feedback alignment (Lillicrap et al., 2016). This assumption states that, for infinitely wide neural networks, if we assume the weights used for forward propagation are independent from those used for back-propagation. When used for the computation of Neural Tangent Kernel, this approximation was proven to give the exact computation for standard architectures such as FFNN, CNN and ResNets Yang, 2020.

Lemma 1 (Corollary of Theorem D.1. in (Yang, 2020))**.** *Consider an FFNN with weights* $\mathbf{W}$*. In the limit of infinite width, we can assume that* $\mathbf{W}^T$ *used in back-propagation is independent from* $\mathbf{W}$ *used for forward propagation, for the calculation of Gradient Covariance and NTK.*

This result has been extensively used in the literature as an approximation before being proved to yield exact computation for gradient covariance and NTK.

Gradient Covariance back-propagation. Analytical formulas for gradient covariance back-propagation were derived using this result, in (Poole et al., 2016; S. Schoenholz et al., 2017; Hayou et al., 2019; Yang, 2019b; Xiao et al., 2018). Empirical results showed an excellent match for FFNN in S. Schoenholz et al., 2017, for Resnets in Yang, 2019b and for CNN in Xiao et al., 2018.

Neural Tangent Kernel. The Gradient Independence approximation was implicitly used in Jacot et al., 2018 to derive the infinite width Neural Tangent Kernel (See Jacot et al., 2018, Appendix A.1). The authors have found that the infinite width NTK computed with the Gradient Independence approximation yields excellent match with empirical (exact) NTK.

A.3. Back-propagation

For FFNN layers, let $q_l(x) := q_l(x,x)$ be the variance of $z_l^1(x)$ (the choice of the index 1 is not important since, in the infinite width limit, the random variables $(z_l^i(x))_{i\in[1:N_l]}$ are i.i.d). Let $q_l(x,x')$, resp. $c_l^1(x,x')$ be the covariance, resp. the correlation between $z_l^1(x)$ and $z_l^1(x')$. For Gradient back-propagation, let $\tilde{q}_l(x,x')$ be the Gradient covariance defined by $\tilde{q}_l(x,x') = \mathbb{E}\left[\frac{\partial \mathcal{L}}{\partial z_l^1}(x)\frac{\partial \mathcal{L}}{\partial z_l^1}(x')\right]$ where $\mathcal{L}$ is some loss function. Similarly, let $\tilde{q}_l(x)$ be the Gradient variance at point x. We also define $\dot{q}_l(x,x') = 2\mathbb{E}[\phi'(z_{l-1}^1(x))\phi'(y_{l-1}^1(x'))]$.

Given two inputs $x, x' \in \mathbb{R}^d$, using Central Limit Theorem as in S. Schoenholz et al., 2017, we obtain

$$q_l(x,x') = 2\mathbb{E}\left[\phi\left(\sqrt{q_l(x)}Z_1\right)\phi\left(\sqrt{q_l(x')}(c^{l-1}Z_1 + \sqrt{1-(c^{l-1})^2}Z_2\right)\right], \quad Z_1, Z_2 \overset{iid}{\sim} \mathcal{N}(0,1),$$

with $c^{l-1} := c^{l-1}(x,x')$.
With ReLU, and since ReLU is positively homogeneous (i.e. $\phi(\lambda x) = \lambda\phi(x)$ for $\lambda \geq 0$), we have that

$$q_l(x,x') = \sqrt{q^l(x)}\sqrt{q^l(x')}g(c^{l-1})$$

where g is the ReLU correlation function given by Hayou et al., 2020

$$g(c) = \frac{1}{\pi}(c\arcsin c + \sqrt{1-c^2}) + \frac{1}{2}c. \tag{3}$$

Gradient back-propagation. The gradient back-propagation is given by

$$\frac{\partial f_{l:L}}{\partial y_i^l} = \phi'(y_i^l)\sum_{j=1}^{N_{l+1}}\frac{\partial f_{l:L}}{\partial y_j^{l+1}}W_{ji}^{l+1}.$$

where $f_{l:L}$ is the mapping from layer l to the output. Using the Gradient Independence in the infinite width limit (Lemma 1) and assuming all layer widths go to infinity at the same rate, a Central Limit Theorem argument yields (see e.g. Section 7.9 in the appendix in S. Schoenholz et al., 2017)

$$\tilde{q}_l(x,x') = \tilde{q}_{l+1}(x,x')g'(c_l(x,x')),$$

where g is the function defined in Eq. (3).

By telescopic product, we obtain

$$\tilde{q}_l(x,x') = \tilde{q}_L(x,x')\prod_{k=l}^{L-1}g'(c_k(x,x')) = \tilde{q}_L(x,x')\frac{\zeta_L(x,x')}{\zeta_l(x,x')}. \tag{4}$$

where $\zeta_m(x,x') = \prod_{k=1}^{m-1}g'(c_k(x,x'))$ for $m \geq 2$.

A.4. Standard parameterization Vs NTK parameterization

Many papers that study the NTK consider the so-called NTK parameterization given by

$$z_1(x) = \frac{2}{\sqrt{d}}W_1x + b_1 \tag{5}$$

$$z_l(x) = \frac{2}{\sqrt{N_{l-1}}}W_l\phi(z_{l-1}(x)), \quad 2 \leq l \leq L, \tag{6}$$

where the weights W_l^{ij} are initialized with standard normal distribution $\mathcal{N}(0,1)$. However, both parameterizations yield the same quantities for signal propagation at initialization, i.e. the covariance q_l is the same for both parameterizations. In our proofs, we will refer to results in Hayou et al., 2020 and S. Schoenholz et al., 2017 that consider either the NTK or the standard paramaterization.

B. Proofs

B.1. Proof of Theorem 1

Theorem 1 (Forward Information Loss)**.** *Let $\epsilon \in (0,1)$, and consider the set E_ϵ as in Eq. (10). Define $I^f_{l,\infty}(x,x') := \lim_{N\to\infty} I^f_{l,N}(x,x')$ for all $x, x' \in \mathbb{R}^d$. We have that*

$$\sup_{(x,x')\in E_\epsilon} |I^f_{l,\infty}(x,x') - 1/2| = \Theta(l^{-2}).$$

Theorem 1 is a corollary of a previous result that appeared in Hayou et al., 2020. The proof techniques for the latter rely on an asymptotic analysis of a well defined covariance kernel in the limit of large l, coupled with a uniform bounding of the convergence rate.

Proof. Fix $(x,x') \in E$. From Appendix A, we know that $I^f_{l,\infty}(x,x') = \frac{1}{2} q_l(x,x')$ where q_l is the covariance between $z^1_l(x), z^1_l(x')$ given by

$$q_l(x,x') = \mathbb{E}[z^1_l(x) z^1_l(x')]$$

Since $q_1(x,x) = q_1(x',x') = 1$, $q_1(x,x')$ can be seen as the correlation between $z^1_1(x)$ and $z^1_1(x')$. Recursively, it is straightforward that $q_l(x,x) = q_l(x',x') = 1$ for all l, suggesting that $q_l(x,x')$ can be seen as the correlation between $z^1_l(x)$ and $z^1_l(x')$.

From 'Appendix Lemma 1' in Hayou et al., 2020, we have that

$$\sup_{(x,x')\in E_\epsilon} |q_l(x,x') - 1| = \Theta(l^{-2})$$

which yields the desired result. □

B.2. Proof of Theorem 2

We first prove a result that will be useful in the proof of Theorem 2.

Lemma 2 (Uniform Asymptotic Expansion)**.** *Let $a \geq 1$ be a positive integer. We define the sequence $(b_l)_{l\geq 0}$ by*

$$b_l = \beta_l b_{l-1},$$

where $(\beta_l)_{l\geq 0}$ is a sequence of reals numbers that satisfy $\beta_l = 1 - \frac{a}{l} + \kappa \frac{\log(l)}{l^2} + \mathcal{O}(l^{-2})$ where $\kappa \neq 0$ is a constant that does not depend on β_0. Assume that the $\mathcal{O}$ bound is uniform over β_0. Then, uniformly over β_0, we have that

$$\log(b_l) = -a\log(l) + \kappa \frac{\log(l)}{l} + \mathcal{O}(l^{-1})$$

Proof. Let $r_l := b_l l^a$. We have that

$$\begin{aligned} r_l &= b_l r_{l-1}(1 + a l^{-1} + \mathcal{O}(l^{-2})) \\ &= (1 + \kappa \frac{\log(l)}{l^2} + \mathcal{O}(l^{-2})) r_{l-1} \end{aligned}$$

which yields

$$\log(r_l / r_{l-1}) = \kappa \frac{\log(l)}{l^2} + \mathcal{O}(l^{-2})$$

Since the series on the right side converge, we have that

$$\begin{aligned}\sum_{k\geq l}\log(r_k/r_{k-1}) &= \sum_{k\geq l}\kappa\frac{\log(k)}{k^2}+\mathcal{O}(\sum_{k\geq l}k^{-2})\\ &= -\kappa\frac{\log(l)+1}{l}+\mathcal{O}(\log(l)l^{-2})+\mathcal{O}(l^{-1})\\ &= -\kappa\frac{\log(l)}{l}+\mathcal{O}(l^{-1})\end{aligned}$$

□

where we have use the integral estimates of the remainders of series. Since the $\mathcal{O}$ bound in β_l is uniform over β_0 by assumption, then the resulting $\mathcal{O}$ bound in $\log(r_l)$ is also uniform over β_0, which concludes the proof.

Theorem 2 (Backward Information Loss)**.** *Let $\epsilon\in(0,1)$, and consider the set E_ϵ as in Eq. (10). Define $I^b_{l,\infty}(x,x') := \lim_{N\to\infty} I^b_{l,N}(x,x')$ for all $x,x'\in\mathbb{R}^d$. Then,*

- *If $l=\lfloor\alpha L\rfloor$ where $\alpha\in(0,1)$ is a constant, then there exists a constant μ such that in the limit $L\to\infty$,*

$$\sup_{(x,x')\in E_\epsilon}|I^b_{l,\infty}(x,x')-\mu| = \Theta(\log(L)L^{-1})$$

- *In the limit $l,L\to\infty$ with $l/L\to 0$,*

$$\sup_{(x,x')\in E_\epsilon}|I^b_{l,\infty}(x,x')| = \Theta((L/l)^{-3}).$$

Proof. Let $\epsilon\in(0,1)$. Note that $I^l_{b,\infty}(x,x')=\tilde{q}_l(x,x')$ where $\tilde{q}_l$ is defined in Appendix A.3.

Using a Taylor expansion of g near 1, Appendix Lemma 1 in Hayou et al., 2020 shows that there exists a constant $\kappa>0$ such that

$$\sup_{(x,x')\in E}|g'(c_l(x,x'))-1+\frac{3}{l}-\kappa\frac{\log(l)}{l^2}| = \mathcal{O}(l^{-2})$$

Let $\zeta_l(x,x')=\prod_{k=1}^{l-1}g'(c_k(x,x'))$ as in Appendix A.3. It is clear that (ζ_l) satisfies the conditions in Lemma 2. Hence, letting $r_l=\zeta_l(x,x')\,l^3$ [10], we obtain

$$\log(r_l)=\kappa\frac{\log(l)}{l}+\mathcal{O}(l^{-1})$$

where the $\mathcal{O}$ bound is uniform over $(x,x')\in E$.

The loss function is given by $\mathbb{L}(f(x),y)$, therefore $\tilde{q}_L(x,x')=\mathbb{E}[\frac{\partial\mathbb{L}(f(x),y)}{\partial z^1_L(x)}\frac{\partial\mathbb{L}(f(x'),y')}{\partial z^1_L(x')}]$. Using the result on the correlation propagation from Appendix Lemma 1 in Hayou et al., 2020, we obtain that $\tilde{q}_L(x,x')=\tilde{q}+\mathcal{O}(L^{-2})$ as L goes to infinity, where $\tilde{q}$ is independent of (x,x').

Now let us discuss the two cases:

- Case 1 ($l=\lfloor\alpha L\rfloor$): in this case, we have that $\log(r_L)-\log\big(r_{\lfloor\alpha L\rfloor}\big)=\Theta(\log(L)L^{-1})$, which yields

$$\frac{\zeta_L(x,x')}{\zeta_{\lfloor\alpha L\rfloor}(x,x')}=\alpha^3+\Theta(\log(L)L^{-1})$$

  where Θ is uniform over x,x'. This proves that

$$\sup_{(x,x')\in E}\big|\tilde{q}_{\lfloor\alpha L\rfloor}(x,x')-\alpha^3\tilde{q}\big| = \Theta(\log(L)L^{-1})$$

[10] We omit (x,x') to alleviate the notations.

- Case 2 ($l/L \to 0$): in this case, we have that

$$\frac{\zeta_L(x,x')}{\zeta_l(x,x')} \sim (L/l)^{-3},$$

uniformly over x, x'. We conclude that

$$\sup_{(x,x')\in E} |\tilde{q}_l(x,x')| = \Theta((L/l)^{-3}).$$

□

B.3. Proof of Corollary 1

Corollary 1 (Equilibrium)**.** *Under the conditions of Theorems 1 and 2, the equilibrium for an FFNN is ahieved for layers with index*

$$l = \Theta(L^{3/5})$$

where L is the network depth.

Proof. We have two cases:

- If l is of the same order as L, or simply $l = \lfloor \alpha L \rfloor$ where $\alpha \in (0,1)$ is a constant, then to have the equilibrium property, we need to have $l^{-2} = \Theta(\log(L)L^{-1})$ which is absurd. Hence, equilibrium cannot be achieved in this case.
- Therefore, the only possible scenario where equilibrium can be achieved is when $l/L \to 0$, in this case, the equilibrium property implies $l^{-2} = \Theta((L/l)^{-3})$ which yields

$$l = \Theta(L^{3/5})$$

□

B.4. Connection between the equilibrium property and effective parameter updates

The EH conjectures architectural advantage for some FFNN layers to more effectively align with data. Using an approximate analysis of the second order geometry of DNNs, we depict the mechanisms by which the equilibrium property impacts how the alignment evolves during early training. We emphasize that the analysis in this section is not intended to be mathematically rigorous but rather insightful for future work on the EH. We restrict our analysis to the setting of FFNN but we now consider a classification task with k classes (i.e. $o = k$). The loss function $\mathbb{L}$ is the cross-entropy loss. We denote by $F = (f_\theta(x_1)^T, f_\theta(x_2)^T, \dots, f_\theta(x_n)^T) \in \mathbb{R}^{nk}$ the concatenation of all outputs $f_\theta(x_i)$ evaluated on the training dataset, $w = \nabla_F \mathcal{L} \in \mathbb{R}^{nk}$ the gradient of the loss w.r.t to F, $Y \in \mathbb{R}^{nk}$ the concatenation of all one-hot vectors given by labels y_i in the dataset, and $\tilde{Y} = \mathbf{C}Y$ the centered version of Y.

Early training steps. As shown in Fig. 2, the alignments $(A_l)_{1\le l\le L}$ sharply increase at early stages of training (≈ 2 epochs of batch training), and plateau soon afterwards. We hence focus on the evolution of the alignment at early training (let T denote the total number of training epochs).

For ReLU activation (and more generally piecewise linear activations), we can express the gradient updates in terms of the output hessian matrix.

Theorem A. *Gradient updates are given by*

$$\theta(t+1) = \left(\mathbf{I} - \frac{\eta}{L-1}\mathbf{H}_w(t)\right)\theta(t) \tag{7}$$

where, for $v \in \mathbb{R}^{nk}$, $\mathbf{H}_v = \sum_{x\in\mathcal{D}}\sum_{i=1}^{k} v_{x,i}\mathbf{H}^i(x)$, and $\mathbf{H}^i(x) = \frac{\partial^2 f_\theta^i(x)}{\partial\theta^2}$ is the output hessian evaluated at x. For $v_{x,i}$, x indexes the datapoint and i the component.

Proof. Let $\mathbf{B}_{l_1,l_2}$ be the block in $\mathbf{H}^i(x)$ containing all entries of the form $\frac{\partial^2 f^i(x)}{\partial W_{l_1}^{jk} \partial W_{l_2}^{st}}$ where $W_{l_1}^{jk}$ is one of layer l_1's parameters, and $W_{l_2}^{st}$ one of layer l_2's parameters.

If $l_1 = l_2$, by piecewise linearity of ϕ, $\mathbf{B}_{l_1,l_2}$ contains all zero entries.

If $l_1 > l_2$, fixing $W_{l_1}^{jk}$,

$$\frac{\partial^2 f^i(x)}{\partial W_{l_1}^{jk} \partial W_{l_2}^{st}} = \frac{\partial \phi(z_{l1-1}^k(x))}{\partial W_{l_2}^{st}} \frac{\partial f^i(x)}{\partial z_{l_1}}[j]$$

where $\frac{\partial f^i(x)}{\partial z_{l_1}}[j]$ is the j-th entry of $\frac{\partial f^i(x)}{\partial z_{l_1}}$. Hence

$$\sum_{s,t} \frac{\partial^2 f^i(x)}{\partial W_{l_1}^{jk} \partial W_{l_2}^{st}} W_{l_2}^{st} = \left(\sum_{s,t} \frac{\partial \phi(z_{l1-1}^k(x))}{\partial W_{l_2}^{st}} W_{l_2}^{st} \right) \frac{\partial f^i(x)}{\partial z_{l_1}}[j]$$

By piecewise linearity of activation function,

$$\sum_{s,t} \frac{\partial^2 f^i(x)}{\partial W_{l_1}^{jk} \partial W_{l_2}^{st}} W_{l_2}^{st} = \phi(z_{l1-1}^k(x)) \frac{\partial f^i(x)}{\partial z_{l_1}}[j] = \frac{\partial f^i(x)}{\partial W_{l_1}^{jk}}$$

If $l_1 < l_2$, fixing $W_{l_1}^{jk}$,

$$\frac{\partial^2 f^i(x)}{\partial W_{l_1}^{jk} \partial W_{l_2}^{st}} = \phi(z_{l1-1}^k(x)) \frac{\partial \frac{\partial f^i(x)}{\partial z_{l_1}}[j]}{\partial W_{l_2}^{st}}$$

Using piecewise linearity of activation function again, we get:

$$\sum_{s,t} \frac{\partial^2 f^i(x)}{\partial W_{l_1}^{jk} \partial W_{l_2}^{st}} W_{l_2}^{st} = \phi(z_{l1-1}^k(x)) \left(\sum_{s,t} \frac{\partial \frac{\partial f^i(x)}{\partial z_{l_1}}[j]}{\partial W_{l_2}^{st}} W_{l_2}^{st} \right) = \phi(z_{l1-1}^k(x)) \frac{\partial f^i(x)}{\partial z_{l_1}}[j] = \frac{\partial f^i(x)}{\partial W_{l_1}^{jk}}$$

Combining the above results we get: (fixing $l_1 \in \{1, .., L\}$, for any $W_{l_1}^{jk}$)

$$\sum_{l_2=1}^{L} \sum_{s,t} \frac{\partial^2 f^i(x)}{\partial W_{l_1}^{jk} \partial W_{l_2}^{st}} W_{l_2}^{st} = (L-1) \frac{\partial f^i(x)}{\partial W_{l_1}^{jk}}$$

The left hand side is the entry of $\mathbf{H}^i(x)\theta$ corresponding to parameter $W_{l_1}^{jk}$, the right hand side is the entry of $(L-1)\Psi^i(x)$ corresponding to parameter $W_{l_1}^{jk}$. We obtain

$$\Psi^i(x) = \frac{1}{L-1} \mathbf{H}^i(x)\theta$$

By this result, the gradient of loss w.r.t parameters can be written as:

$$\nabla_\theta \mathcal{L} = \left(\frac{\partial \mathcal{L}}{\partial F} \frac{\partial F}{\partial \theta} \right)^T = (w^T \mathbf{\Psi}^T)^T = \mathbf{\Psi} w = \frac{\eta}{L-1} \mathbf{H}_w(t)\theta(t)$$

Hence, gradient updates are given by

$$\theta(t+1) = \left(\mathbf{I} - \frac{\eta}{L-1} \mathbf{H}_w(t) \right) \theta(t)$$

□

Theorem A provides a geometric interpretation of gradient updates. The directions in parameter space where the largest updates occur are controlled by $\mathbf{H}_w$. A similar result holds for tangent features evolution during early training stages. Let us first introduce a key approximation.

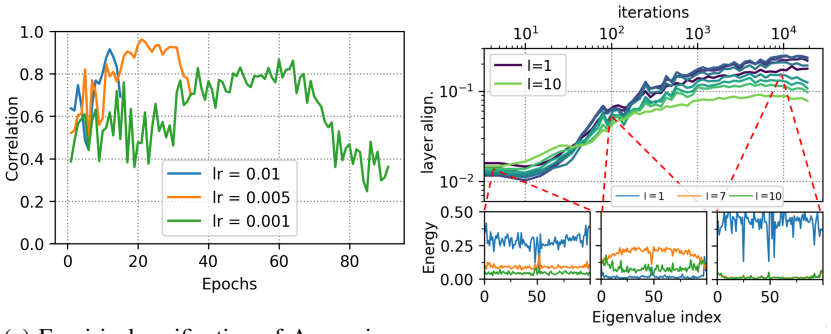
Approximation 1 (Collinearity at early training stages)**.** *At early stages of training, $\tilde{Y}$ and w are almost negatively co-linear. Specifically, $w \approx -\frac{\|w\|}{\|\tilde{Y}\|}\tilde{Y}$. As a result vectors $\mathbf{\Psi}(t+1)\tilde{Y} - \mathbf{\Psi}(t)\tilde{Y}$ and $-\eta \mathbf{H}_w(t)\mathbf{\Psi}(t)\tilde{Y}$ are highly correlated.*

See Appendix B.5 for a theoretical justification of Approximation 1 with empirical evidence of its validity.

Theorem A and Approximation 1 provides a link between the EH and effective feature learning. In Section 3, we explained our intuition on how efficient parameter updates take place in layers at information equilibrium. On the other hand, Theorem A and Approximation 1 suggest that the directions with the largest updates in parameters θ are highly correlated with the directions for the largest updates in $\mathbf{\Psi}\tilde{Y}$. Hence, $\mathbf{\Psi}_l\tilde{Y}$ are updated more effectively at layers with information equilibrium, leading to larger alignment value.

To confirm this intuition, we train an depth 10 FFNN on CIFAR10 and show in Fig. 5b the norm of the top 100 eigenvectors of $\mathbf{H}_w$ (corresponding to top 100 eigenvalues in absolute value) projected to 3 layers, extreme layer 1 and 10 and middle layer 7 (by projection to layer l we refer to the truncation of the vector to leave just the sub-vector that corresponds to layer l). During the sharp increase phase in alignments (middle subfigure in Fig. 5b), the top eigenvectors of the Hessian are more concentrated on intermediate layers, suggesting that the sharpest increase in alignment occurs in those intermediate layers.

Remark. The hessian of the loss is given by $\frac{\partial^2 \mathcal{L}}{\partial \theta^2} = \mathbf{H}_w + \mathbf{\Psi}\frac{\partial^2 \mathcal{L}}{\partial F^2}\mathbf{\Psi}^T := S + I$. Previous works on second order geometry of DNNs (e.g. Ghorbani et al., 2019; Karakida et al., 2019) focused on large positive eigenvalues of the loss hessian arising mostly from I. Approximation 1 shows that at the other end of the spectrum, large negative eigenvalues arising from S influences feature learning in 10. In particular, eigenvectors of S associated with large negative eigenvalues are directions of significant increase in alignment between features and labels.



(a) Empirical verification of Approximation 1

(b) Alignment hierarchy during training

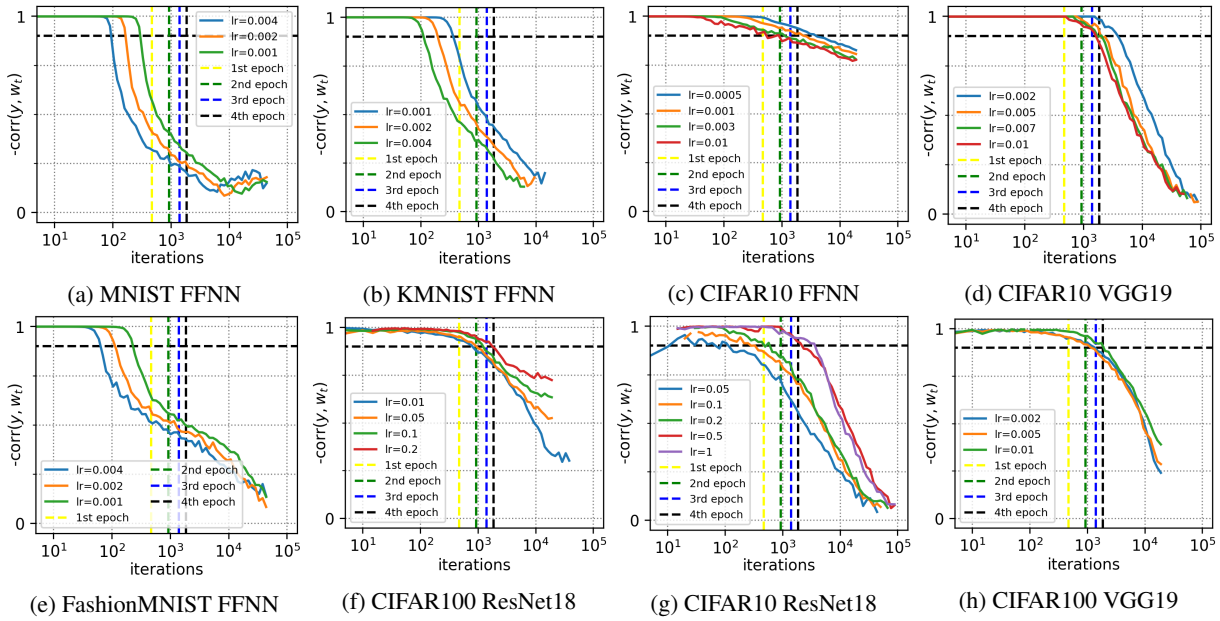
Figure 5: **(a)** Correlation between vectors $U\left(\mathbf{\Psi}(t+1)\tilde{Y} - \mathbf{\Psi}(t)\tilde{Y}\right)$ and $-U\mathbf{H}_w(t)\mathbf{\Psi}(t)\tilde{Y}$ for CIFAR10 on a 10-layer FFNN. **(c)** The norm of the eigenvectors of $\mathbf{H}_w$ (energy in the plot) related to top 100 eigenvalues in absolute value projected to 3 layers: 1, 7, 10 at three training times. The top eigenvalues in absolute value of $\mathbf{H}_w$ are 2, 15, 6 resp ($\mathbf{H}_w$ has a symmetric eigenspectrum shown in Geiger et al., 2019) for three training times. The plot provides an illustration of why alignment hierarchy arises as more energy of top eigenvectors concentrate on intermediate layers during critical increase in alignments.

B.5. Justification of Approximation 1

Approximation 1 (Collinearity at early training stages) *(1) At early stages of training, $\tilde{Y}$ and w are almost negatively co-linear. Specifically, $w \approx -\frac{\|w\|}{\|\tilde{Y}\|}\tilde{Y}$. (2) As a result vectors $\mathbf{\Psi}(t+1)\tilde{Y} - \mathbf{\Psi}(t)\tilde{Y}$ and $-\eta \mathbf{H}_w(t)\mathbf{\Psi}(t)\tilde{Y}$ are highly correlated.*

B.5.1. Justification of (1)

The intuition behind Approximation 1 has roots in the assumption that the dataset is balanced. To see this, let $(x, y) \in \mathcal{D}$ and $\mathbb{L}_x = \mathbb{L}(f_\theta(x), y)$ the cross-entropy loss for the datapoint (x, y). Let c be the true label of x, and $i \in \{1, .., k\}$ such that



(a) MNIST FFNN (b) KMNIST FFNN (c) CIFAR10 FFNN (d) CIFAR10 VGG19

(e) FashionMNIST FFNN (f) CIFAR100 ResNet18 (g) CIFAR10 ResNet18 (h) CIFAR100 VGG19

Figure 6: Empirical validation of Approximation 1: a demonstration that $-\text{corr}(\tilde{Y}, w_t) \approx 1$ at early t.

$i \neq c$. We have that

$$\begin{aligned} \nabla_{f^i_\theta(x)} \mathcal{L} &= \frac{\exp\left(f^i_\theta(x)\right)}{\sum_{j=1}^k \exp\left(f^j_\theta(x)\right)}, \quad i \neq c \\ \nabla_{f^c_\theta(x)} \mathcal{L} &= -\sum_{i \neq c} \frac{\exp\left(f^i_\theta(x)\right)}{\sum_{j=1}^k \exp\left(f^j_\theta(x)\right)} \end{aligned} \tag{8}$$

where $f^i_\theta(x)$ is the i-th entry of $f_\theta(x)$. When the dataset is balanced, i.e. the number of datapoint per class is approximately the same for all classes, the corresponding entries of $\tilde{Y}$ satisfy $\tilde{Y}_{x,c} \approx \frac{k-1}{k}$ and $\tilde{Y}_{x,i} \approx \frac{-1}{k}$. At initialization, with random weights, $f^i_\theta(x)$ are on average similar across choices of x and i. Hence, using Eq. (8), on average we have $w_{x,c} \approx -\frac{k-1}{k}$ and $w_{x,i} \approx \frac{1}{k}$, thus, $\tilde{Y}$ are almost negatively co-linear. Fig. 6 illustrates this result on various architectures and datasets (see also Appendix E). We observe that Approximation 1 also holds during early training steps ($\text{corr}(w, \tilde{Y}) \approx -1$ during the first training epoch).

B.5.2. Justification of (2)

(2) $\mathbf{\Psi}(t+1)\tilde{Y} - \mathbf{\Psi}(t)\tilde{Y}$ and $-\eta \mathbf{H}_w(t)\mathbf{\Psi}(t)\tilde{Y}$ are highly correlated.

We will first need Approximation 2.

Approximation 2 (1^{st} order approximation)**.**

$$\Psi^i(x)(t+1) - \Psi^i(x)(t) = \mathbf{H}^i(x)(t)\left(\theta(t+1) - \theta(t)\right) + \mathcal{E}_{x,i}(t)$$

where $\mathcal{E}_{x,i}(t)$ includes higher order terms of $\theta(t+1) - \theta(t)$. We will first justify Approximation 2:

$$\begin{aligned} \Psi^i(x)(t+1) - \Psi^i(x)(t) &= \left.\frac{\partial f^i(x)}{\partial \theta}\right|_{\theta(t+1)} - \left.\frac{\partial f^i(x)}{\partial \theta}\right|_{\theta(t)} \\ &= \left.\frac{\partial^2 f^i(x)}{\partial \theta^2}\right|_{\theta(t)} \left(\theta(t+1) - \theta(t)\right) + \mathcal{E}_{x,i}(t) \\ &= \mathbf{H}^i(x)(t)\left(\theta(t+1) - \theta(t)\right) + \mathcal{E}_{x,i}(t) \end{aligned}$$

The second step is by Taylor expanding $\frac{\partial f^i(x)}{\partial \theta}\Big|_{\theta(t+1)}$ around $\theta(t)$. Given Approximation 2, for an arbitrary fixed vector $v \in \mathbb{R}^{kn}$, the evolution of $\mathbf{\Psi} v$ can be approximated by:

$$\begin{aligned}
\mathbf{\Psi}(t+1)v - \mathbf{\Psi}(t)v &= \sum_{x\in\mathcal{D}}\sum_{j=1}^{k} v_{x,i}\left(\mathbf{H}^i(x)(t)\left(\theta(t+1)-\theta(t)\right)+\mathcal{E}_{x,i}(t)\right) \\
&= -\eta\sum_{x\in\mathcal{D}}\sum_{j=1}^{k} v_{x,i}\mathbf{H}^i(x)(t)\mathbf{\Psi}(t)w + \sum_{x\in\mathcal{D}}\sum_{j=1}^{k} v_{x,i}\mathcal{E}_{x,i}(t) \\
&= -\eta\mathbf{H}_v(t)\mathbf{\Psi}(t)w + \sum_{x\in\mathcal{D}}\sum_{j=1}^{k} v_{x,i}\mathcal{E}_{x,i}(t)
\end{aligned} \tag{9}$$

Setting $v = \tilde{Y}$ we get:

$$\begin{aligned}
\mathbf{\Psi}(t+1)\tilde{Y} &= \mathbf{\Psi}(t)\tilde{Y} - \eta\mathbf{H}_{\tilde{Y}}\mathbf{\Psi}(t)w + \sum_{x\in\mathcal{D}}\sum_{j=1}^{k}\tilde{Y}_{x,i}\mathcal{E}_{x,i}(t) \\
&\approx \mathbf{\Psi}(t)\tilde{Y} - \eta\mathbf{H}_w\mathbf{\Psi}(t)\tilde{Y} + \sum_{x\in\mathcal{D}}\sum_{j=1}^{k}\tilde{Y}_{x,i}\mathcal{E}_{x,i}(t) \\
&= \left(\mathbf{I}-\eta\mathbf{H}_w\right)\mathbf{\Psi}(t)\tilde{Y} + \sum_{x\in\mathcal{D}}\sum_{j=1}^{k}\tilde{Y}_{x,i}\mathcal{E}_{x,i}(t)
\end{aligned} \tag{10}$$

In the second step we use Approximation 1 (1). As $\sum_{x\in\mathcal{D}}\sum_{j=1}^{k}\tilde{Y}_{x,i}\mathcal{E}_{x,i}(t)$ contains high order terms in η, this is small compared to $\left(\mathbf{I}-\eta\mathbf{H}_w\right)\mathbf{\Psi}(t)\tilde{Y}$. Hence, the update in feature vector $\mathbf{\Psi}(t+1)\tilde{Y} - \mathbf{\Psi}(t)\tilde{Y}$ is highly correlated with $-\eta\mathbf{H}_w\mathbf{\Psi}(t)\tilde{Y}$.

C. A Justification of the choice of Tangent Features and the CKA alignment

Let $(x,y) \in \mathcal{D}$ and $\mathbb{L}_x = \mathbb{L}(f_\theta(x), y)$ the cross-entropy loss for the datapoint (x,y). Let c be the true label of x, and $i \in \{1,..,k\}$ such that $i \neq c$. We have that

$$\begin{aligned}
\nabla_{f^i_\theta(x)}\mathbb{L} &= \frac{\exp\left(f^i_\theta(x)\right)}{\sum_{j=1}^{k}\exp\left(f^j_\theta(x)\right)}, \quad i \neq c \\
\nabla_{f^c_\theta(x)}\mathbb{L} &= -\sum_{i\neq c}\frac{\exp\left(f^i_\theta(x)\right)}{\sum_{j=1}^{k}\exp\left(f^j_\theta(x)\right)}
\end{aligned} \tag{11}$$

Hence, for any datapoint (x,y), we have that

$$\nabla_{f_\theta(x)}\mathbb{L} = \text{softmax}(f(x)) - y.$$

It is straightforward that update of the network output (evaluated on the whole dataset (X,Y)) with one gradient step is given by (see e.g. Jacot et al., 2018)

$$df_t(X) = -\eta\hat{\mathbf{K}}\nabla_y\mathbb{L}(f_t(X),Y),$$

where $\hat{\mathbf{K}}^L$ is the tangent kernel matrix, $f_t(X), Y \in \mathbb{R}^{on}$ are the concatenated vectors of $(f(x_i))_{1\leq i\leq n}$ and $(y)_{1\leq i\leq n}$, and $\eta = \text{LR}/n$ is the normalized learning rate. Consider the case of large width $N \gg 1$. At initialization, on average, the network output is a random classifier.

$$\mathbb{E}_W\left[\frac{d}{dt}f_t(X)\right] \approx \hat{\mathbf{K}}CY,$$

where we have used the gradient independence result from Lemma 1 and the approximation that the NTK is almost deterministic in the large width limit (see Jacot et al., 2018). This yields

$$\mathbb{E}_W\left[\frac{d}{dt}y^T f_t(X)\right] \approx y^T \hat{\mathbf{K}} y.$$

The alignment between the tangent kernel and data labels has a direct impact on how fast the network output aligns with the true labels.

To measure the role played by each layer, let us consider the scenario when we freeze all but the l^{th} layer parameters, the previous dynamics become

$$\mathbb{E}_W\left[\frac{d}{dt}y^T \tilde{f}_t(X)\right] \approx y^T \hat{\mathbf{K}}_l y,$$

where $\hat{\mathbf{K}}_l$ is the tangent kernel matrix for layer l. Observe that

$$y^T \hat{\mathbf{K}}_l y \leq \rho(\hat{\mathbf{K}}_l)\|y\|^2 \leq \text{Tr}(\hat{\mathbf{K}}_l)\|y\|^2,$$

with equality if and only if the kernel matrix $\hat{\mathbf{K}}_l$ is perfectly aligned with the data labels matrix yy^T. Therefore, the alignment A_l has a direct impact on the alignment between the data labels and output function (note that a perfect alignment between y and $f_t(X)$ indicates 100% classification accuracy).

D. Optimal Feature Evolution Scheme

Shan and Bordelon (2021) proposed Optimal Feature Evolution (OFE) scheme to model the evolution of tangent features during gradient descent (GD) training. Under OFE, the tangent features $\mathbf{\Psi}$ evolve greedily so that the change in empirical loss $\mathcal{L}$ is maximised at each time step. However, it is not verified empirically if OFE matches any variants of GD methods. In the following, we propose Generalised Optimal Feature Evolution (GOFE) scheme to capture GD methods more closely and gain insights into the evolution of layerwise CKA.

D.1. Optimal Feature Evolution with fixed learning rates

In Shan and Bordelon, 2021, the optimal feature evolution paradigm is only given for MSE loss. The following is a summary of OFE evolution scheme for any twice differentiable loss $\mathcal{L}$. We inherit notation from Appendix B.

By GD training dynamics we have:

$$\begin{aligned}\frac{\partial \theta^T}{\partial t} &= -\eta\frac{\partial \mathcal{L}}{\partial \theta} = -\eta\frac{\partial \mathcal{L}}{\partial F}\frac{\partial F}{\partial \theta} \\ &= -\eta w^T \mathbf{\Psi}^T \\ \Rightarrow \frac{\partial \theta}{\partial t} &= -\eta \mathbf{\Psi} w\end{aligned} \tag{12}$$

The evolution of $\mathcal{L}$ is:

$$\frac{\partial \mathcal{L}}{\partial t} = \frac{\partial \mathcal{L}}{\partial F}\frac{\partial F}{\partial \theta}\frac{\partial \theta}{\partial t} = -\eta w^T \mathbf{\Psi}^T \mathbf{\Psi} w \tag{13}$$

Using the same argument as in Shan and Bordelon, 2021, we optimise the term $w^T \mathbf{\Psi}^T \mathbf{\Psi} w$ w.r.t $\mathbf{\Psi}$ by evolving $\mathbf{\Psi}$ in the direction of largest decrease in $-w^T \mathbf{\Psi}^T \mathbf{\Psi} w$ with a learning rate of λ. This yield:

$$\begin{aligned}\frac{\partial \mathbf{\Psi}^T}{\partial t} &= -\lambda\frac{\partial\left(-\eta w^T \mathbf{\Psi}^T \mathbf{\Psi} w\right)}{\partial \mathbf{\Psi}} \\ \Rightarrow \frac{\partial \mathbf{\Psi}}{\partial t} &= 2\lambda\eta \mathbf{\Psi} w w^\top\end{aligned} \tag{14}$$

We could absorb the 2 factor in this equation into λ to produce the dynamics:

$$\begin{aligned}\frac{\partial \mathbf{\Psi}}{\partial t} &= \lambda \eta \mathbf{\Psi} w w^{\top} \\ \frac{\partial w}{\partial t} &= -\eta \frac{\partial w}{\partial \theta} \mathbf{\Psi}^T \mathbf{\Psi} w\end{aligned} \tag{15}$$

At a first look, this model adds an interesting layer of complexity over fixed tangent kernel learning: the kernel evolves in predictable and alignment-boosting ways. However, there's no strong empirical evidence that OFE captures any variant of GD training.

D.2. Generalised Optimal Feature Evolution and GD training

To better capture GD/SGD/NGD dynamics, we first introduce the paradigm of Generalised Optimal Feature Evolution (GOFE):

$$\begin{aligned}\frac{\partial \mathbf{\Psi}}{\partial t} &= \eta \mathbf{V} \mathbf{\Psi} w w^T \\ \frac{\partial w}{\partial t} &= -\eta \frac{\partial w}{\partial F} \mathbf{\Psi}^T \mathbf{A} \mathbf{\Psi} w = -\eta \frac{\partial^2 \mathcal{L}}{\partial F^2} \mathbf{\Psi}^T \mathbf{A} \mathbf{\Psi} w\end{aligned} \tag{16}$$

where $\mathbf{V}$ is a velocity vector that may or may not depend on time step t, $\mathbf{A}$ is a time-dependent matrix describing the training procedure used (i.e. for full batch GD, $\mathbf{A}$ is simply the identity matrix, and for natural gradient descent, $\mathbf{A}$ is the inverse of Fisher Information Matrix). $\frac{\partial w}{\partial F}$ gradient of w w.r.t to the output F. The last equality is due to definition of $w^T = \frac{\partial \mathcal{L}}{\partial F}$. Note that in OFE, $\mathbf{V}$ is simply a diagonal matrix with diagonal entries λ. Note that in practice the feature evolution is realised by the set of difference equations:

$$\begin{aligned}\Delta_t(\mathbf{\Psi}) &= \eta \mathbf{V}_t \mathbf{\Psi}_t w_t w_t^T \\ \Delta_t(\theta) &= -\eta \mathbf{A}_t \mathbf{\Psi}_t w_t\end{aligned} \tag{17}$$

This would allow us to conduct several calculations exactly in the following. The t index each time step.

It turns out that for each gradient descent dynamics, be it full batch or stochastic GD or natural gradient descent, there is an equivalent formulation of its training dynamics in terms of GOFE. By 'equivalent' we mean that at each time step, the gradient propagated to the weights of the network is the same. To achieve this equivalence we need the following two dynamics of gradient changes to be the same:

Under GOFE:

$$\begin{aligned}\frac{\partial(\mathbf{\Psi} w)}{\partial t} &= \frac{\partial(\mathbf{\Psi})}{\partial t} w + \mathbf{\Psi} \frac{\partial(w)}{\partial t} \\ &= \eta \mathbf{V} \mathbf{\Psi} w w^T w - \eta \mathbf{\Psi} \frac{\partial^2 \mathcal{L}}{\partial F^2} \mathbf{\Psi}^T \mathbf{A} \mathbf{\Psi} w \\ &= \eta \|w\|^2 \mathbf{V} \mathbf{\Psi} w - \eta \mathbf{\Psi} \frac{\partial^2 \mathcal{L}}{\partial F^2} \mathbf{\Psi}^T \mathbf{A} \mathbf{\Psi} w\end{aligned} \tag{18}$$

Under gradient descent with gradient adjustment matrix $\mathbf{A}$:

$$\begin{aligned}\frac{\partial(\mathbf{\Psi} w)}{\partial t} &= \frac{\partial(\mathbf{\Psi} w)}{\partial \theta} \frac{\partial \theta}{\partial t} = \frac{\partial^2 \mathcal{L}}{\partial \theta^2} (-\eta \mathbf{A} \Psi w) = -\eta \frac{\partial^2 \mathcal{L}}{\partial \theta^2} \mathbf{A} \mathbf{\Psi} w \\ &= -\eta \mathbf{H}_w \mathbf{A} \mathbf{\Psi} w - \eta \mathbf{\Psi} \frac{\partial^2 \mathcal{L}}{\partial F^2} \mathbf{\Psi}^T \mathbf{A} \mathbf{\Psi} w\end{aligned} \tag{19}$$

During the derivation we have used a well known decomposition of the loss hessian:

$$\frac{\partial^2 \mathcal{L}}{\partial \theta^2} = \mathbf{H}_w + \mathbf{\Psi} \frac{\partial^2 \mathcal{L}}{\partial F^2} \mathbf{\Psi}^T \tag{20}$$

where $\mathbf{H}_w$ is the same as $\mathbf{H}_w$ in Appendix B.4. Equating the two dynamics we need:

$$\begin{aligned}&\eta\|w\|^2\mathbf{V}\mathbf{\Psi}w-\eta\mathbf{\Psi}\frac{\partial^2\mathcal{L}}{\partial F^2}\mathbf{\Psi}^T\mathbf{A}\mathbf{\Psi}w=-\eta\mathbf{H}_w\mathbf{A}\mathbf{\Psi}w-\eta\mathbf{\Psi}\frac{\partial^2\mathcal{L}}{\partial F^2}\mathbf{\Psi}^T\mathbf{A}\mathbf{\Psi}w\\&\iff(\|w\|^2\mathbf{V}+\mathbf{H}_w\mathbf{A})\mathbf{\Psi}w=0\end{aligned}\tag{21}$$

We hence set $\mathbf{V}=-\frac{\mathbf{H}_w\mathbf{A}}{\|w\|^2}$. Under the assumption that w and y are highly correlated at early training, we could directly derive the evolution of $\mathbf{\Psi}$. In fact, under GD ($\mathbf{A}$ being identity matrix):

$$\frac{\partial\mathbf{\Psi}}{\partial t}=-\eta\frac{\mathbf{H}_w\mathbf{A}}{\|w\|^2}\mathbf{\Psi}ww^T=-\eta\mathbf{H}_w\mathbf{\Psi}\frac{ww^T}{\|w\|^2}=-\eta\mathbf{H}_w\mathbf{\Psi}\frac{yy^T}{\|y\|^2}\tag{22}$$

We could test GOFE against results derived without it, in fact multiplying y to the above equation yields:

$$\frac{\partial\mathbf{\Psi}y}{\partial t}=-\eta\mathbf{H}_w\mathbf{\Psi}\frac{yy^T}{\|y\|^2}y=-\eta\mathbf{H}_w\mathbf{\Psi}y\tag{23}$$

which is the continuous version of Eq. (10). Also for any fixed vector u orthogonal to y, we have:

$$\frac{\partial\mathbf{\Psi}u}{\partial t}=-\eta\mathbf{H}_w\mathbf{\Psi}\frac{yy^T}{\|y\|^2}u=0\tag{24}$$

This relates to the increase in tangent kernel anistropy in Baratin et al., 2021, as $u^T\mathbf{\Psi}^T\mathbf{\Psi}u$ stays constant over training while $y^T\mathbf{\Psi}^T\mathbf{\Psi}y$ increases sharply due to large negative eigenvalues in $\mathbf{H}_w$.

D.3. Explaining the Hierarchy using feature evolution scheme

Eq. (23) gives us a way to describe $\mathbf{\Psi}(t+1)y$ as $\mathbf{H}(t)\mathbf{\Psi}(t)y$ for some matrix $\mathbf{H}(t)=(\mathbf{I}-\eta\mathbf{H}_w(t))$ which also describes evolution of parameters. Take an orthogonal basis consisting of $u_0=\frac{y}{\|y\|}, u_1=\frac{1}{\sqrt{kN}}(1,1...,1)^T, u_2,...,u_N\in\mathbb{R}^{kN}$, $\mathbf{U}$ be the $kN\times kN$ matrix with u_i as columns and we would have:

$$\begin{aligned}A_l(t+1)&=\frac{y^T\mathbf{\Psi}(t+1)^T\mathbf{M}_l\mathbf{\Psi}(t+1)y}{\|\mathbf{\Psi}(t+1)^T\mathbf{M}_l\mathbf{\Psi}(t+1)\mathbf{C}\|_F\|y\|^2}\\&=A_l(t)\cdot\frac{u_0^T\mathbf{\Psi}(t+1)^T\mathbf{M}_l\mathbf{\Psi}(t+1)u_0}{u_0^T\mathbf{\Psi}(t)^T\mathbf{M}_l\mathbf{\Psi}(t)u_0}\cdot\frac{\|U^T\mathbf{\Psi}(t)^T\mathbf{M}_l\mathbf{\Psi}(t)\mathbf{C}U\|_F}{\|U^T\mathbf{\Psi}(t+1)^T\mathbf{M}_l\mathbf{\Psi}(t+1)\mathbf{C}U\|_F}\\&\approx A_l(t)\cdot\frac{u_0^T\mathbf{\Psi}(t)^T\mathbf{H}(t)^T\mathbf{M}_l\mathbf{H}(t)\mathbf{\Psi}(t)u_0}{u_0^T\mathbf{\Psi}(t)^T\mathbf{M}_l\mathbf{\Psi}(t)u_0}\\&=A_l(t)\cdot\frac{\theta(t)^T\mathbf{H}_y(t)^T\mathbf{H}^T\mathbf{M}_l\mathbf{H}\mathbf{H}_y(t)\theta(t)}{\theta(t)^T\mathbf{H}_y(t)^T\mathbf{M}_l\mathbf{H}_y(t)\theta(t)}\\&\approx A_l(t)\cdot\frac{\operatorname{tr}\left(\mathbf{H}_y(t)^T\mathbf{H}(t)^T\mathbf{M}_l\mathbf{H}(t)\mathbf{H}_y(t)\right)}{\operatorname{tr}\left(\mathbf{H}_y(t)^T\mathbf{M}_l\mathbf{H}_y(t)\right)}\\&\approx A_l(t)\cdot\frac{\operatorname{tr}\left(\mathbf{M}_l\left(\mathbf{I}-\eta\mathbf{H}_w(t)\right)\mathbf{H}_w^2(t)\left(\mathbf{I}-\eta\mathbf{H}_w(t)\right)\right)}{\operatorname{tr}\left(\mathbf{M}_l\mathbf{H}_w^2(t)\right)}\\&=A_l(t)-2\eta\frac{\operatorname{tr}\left(\mathbf{M}_l\mathbf{H}_w^3(t)\right)}{\operatorname{tr}\left(\mathbf{M}_l\mathbf{H}_w^2(t)\right)}+\eta^2\frac{\operatorname{tr}\left(\mathbf{M}_l\mathbf{H}_w^4(t)\right)}{\operatorname{tr}\left(\mathbf{M}_l\mathbf{H}_w^2(t)\right)}\end{aligned}\tag{25}$$

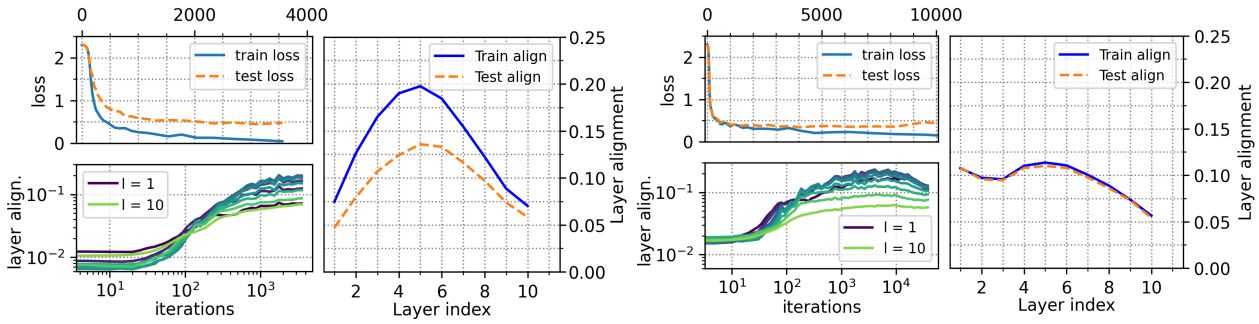
The first approximation holds in the case of large N and the second approximation is based on the assumption that $\theta(t)$ is independent from $\mathbf{H}_y(t)$ and $\mathbf{H}(t)$, and each entry is drawn from i.i.d normal distribution.The second approximation has its roots in Gradient Independence Appendix A.2. The third approximation uses Approximation 1. This derivation illustrates that hierarchical structure of CKA likely arise out of bias in $\mathbf{H}_w$'s third and fourth moment. Actually, for common learning rates of ≈ 0.005 used for deep networks, $\mathbf{H}_w$'s largest positive eigenvalue is usually around $5-15$, hence the third part of

Eq. (25) is dominated by the first and second part which is around $0.01 - 0.07$. We later empirically illustrate interesting structural bias in $\mathbf{H}_w(t)$. Let $\mathbf{V}(t)$ diagonalises $\mathbf{H}_w(t)$:

$$\frac{\operatorname{tr}\left(\mathbf{M}_l\mathbf{H}_w^3(t)\right)}{\operatorname{tr}\left(\mathbf{M}_l\mathbf{H}_w^2(t)\right)} = \frac{\operatorname{tr}\left(\mathbf{V}(t)^T\mathbf{M}_l\mathbf{V}(t)\mathbf{V}(t)^T\mathbf{H}_w^3(t)\mathbf{V}(t)\right)}{\operatorname{tr}\left(\mathbf{V}(t)^T\mathbf{M}_l\mathbf{V}(t)\mathbf{V}(t)^T\mathbf{H}_w^2(t)\mathbf{V}(t)\right)} = \frac{\sum_i c_i(t)\lambda_i^3}{\sum_i c_i(t)\lambda_i^2}$$

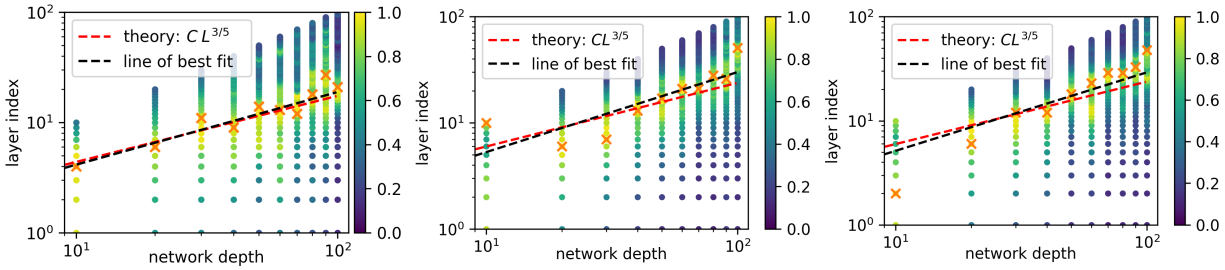
where $c_i(t) := v_i(t)^T\mathbf{M}_l v_i(t)$ and $\lambda_i(t)$ is the eigenvalue corresponding to $v_i(t)$. The quantity is a weighted average of all eigenvalues of $\mathbf{H}_w(t)$.

E. Further experimental results



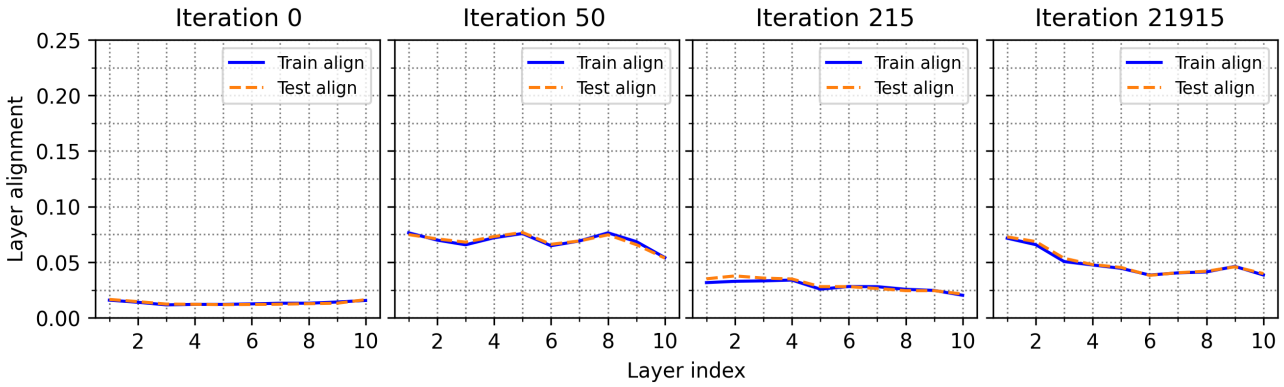
(a) KMNIST (b) Fashion MNIST

Figure 7: Supplementary experiments for Fig. 2. Layerwise alignment hierarchy for the KMNIST and Fashion MNIST datasets when trained on a FFNN with depth 10 and width 256. Left hand panels show progression of loss and layer alignment with iterations of SGD. Right hand panel shows layer alignment at the end of training. Experiments above and in Fig. 2 used 10 layer FFNNs with 256 neurons in each layer, and were optimised with SGD with weight decay, momentum, and learning rates of 0.003.

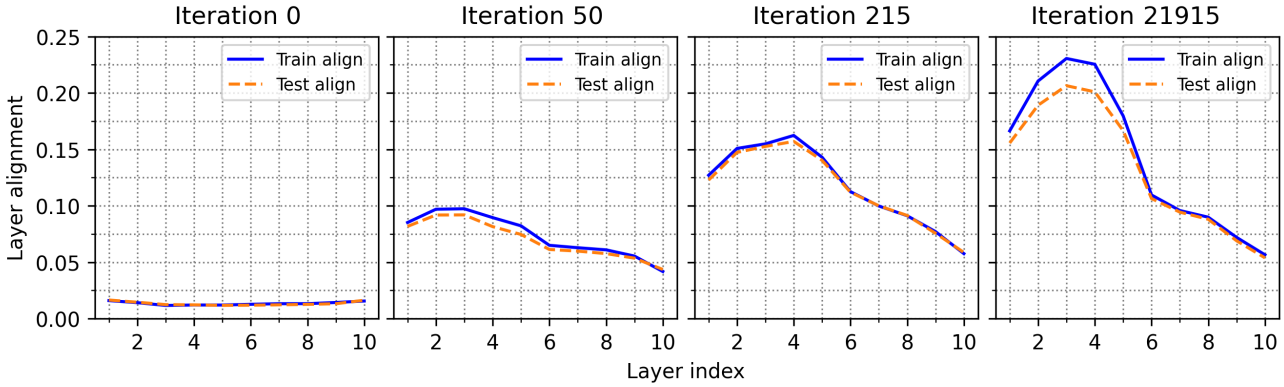


(a) FFNN on Fashion MNIST (b) FFNN on Fashion MNIST (c) FFNN on Fashion MNIST

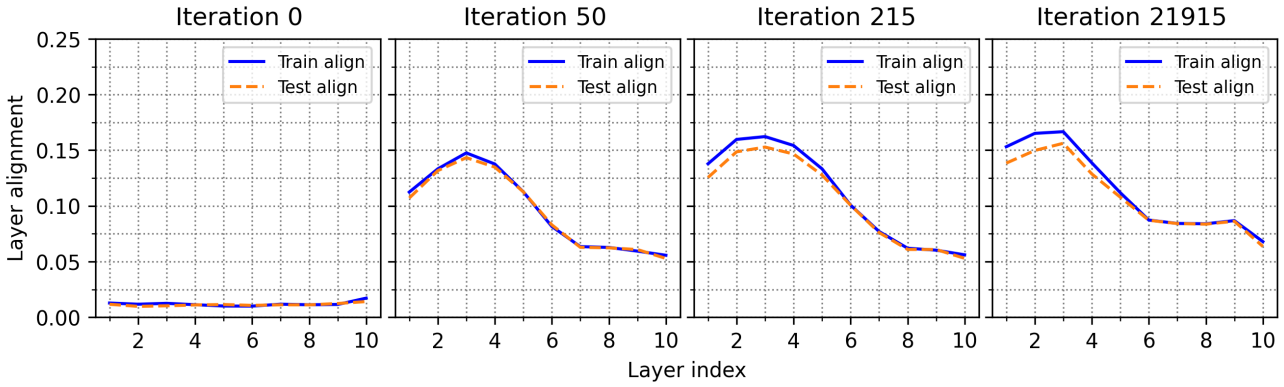
Figure 8: Here we compare (a) Fashion MNIST on a FFNN with batch size 128 and learning rate for the $10j$'th layer given by $[0.003, 0.004, 0.004, 0.002, 0.001, 0.0007, 0.0003, 0.0002, 0.0001, 0.00007]$. This variation of layer-wise learning rate produces the best generalisation at each depth. (b) Fashion MNIST on a FFNN with batch size 128 and learning rate 4x smaller per layer than in (a). (c) uses the same learning rates as (a) but a batch size of 512. Note that (a) is the same experiment as Fig. 3b in the main text. There is a clear upward shift in the y-intercept with decreasing learning rate (b) and increased batch size (c), as discussed in Section 4, meaning that the peak of the AH shifts towards the last layer, and this correlates with poorer performance.



(a) Learning Rate 0.0005, test accuracy 88.4%, test loss 0.53

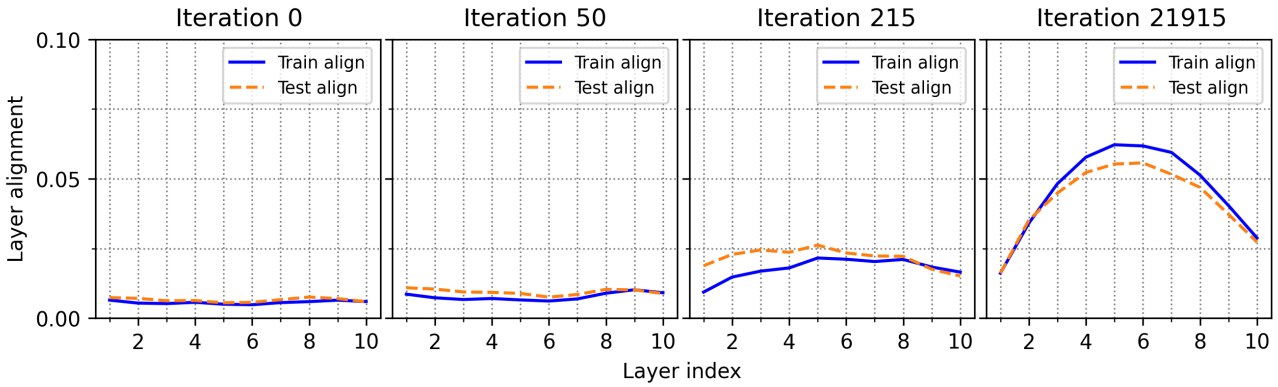


(b) Learning Rate 0.003, test accuracy 88.3%, test loss 0.53

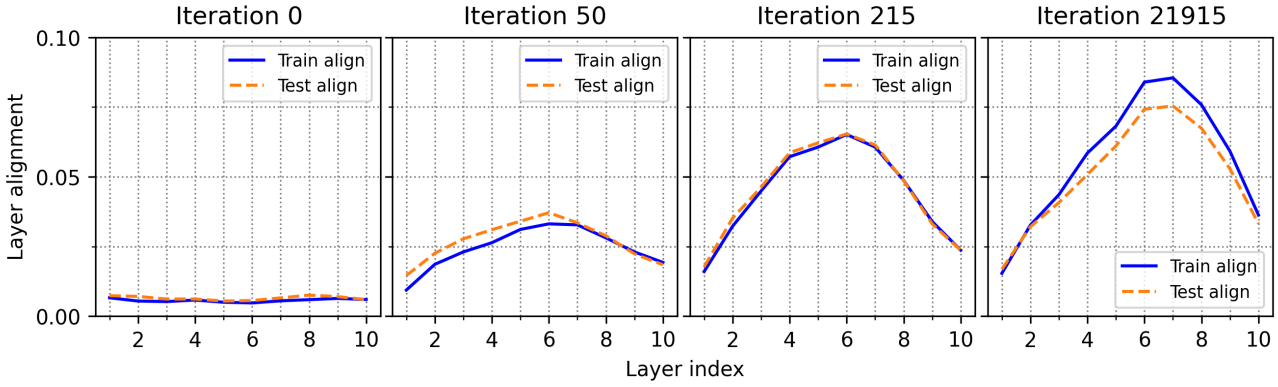


(c) Learning Rate 0.05, test accuracy 87.1%, test loss 0.41

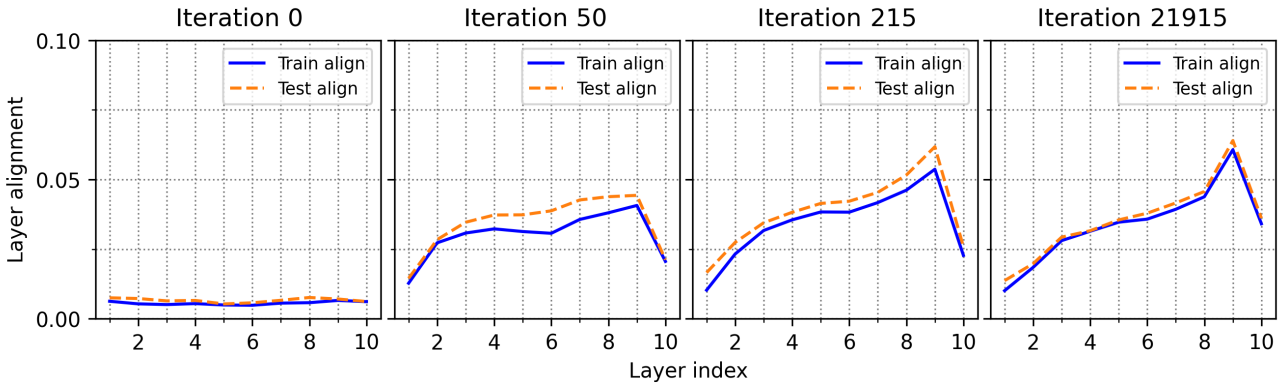
Figure 9: Alignment progress during training. Fashion MNIST. Further detail for Fig. 7b.



(a) Learning Rate 0.0005, test accuracy 53.9%, test loss 1.30

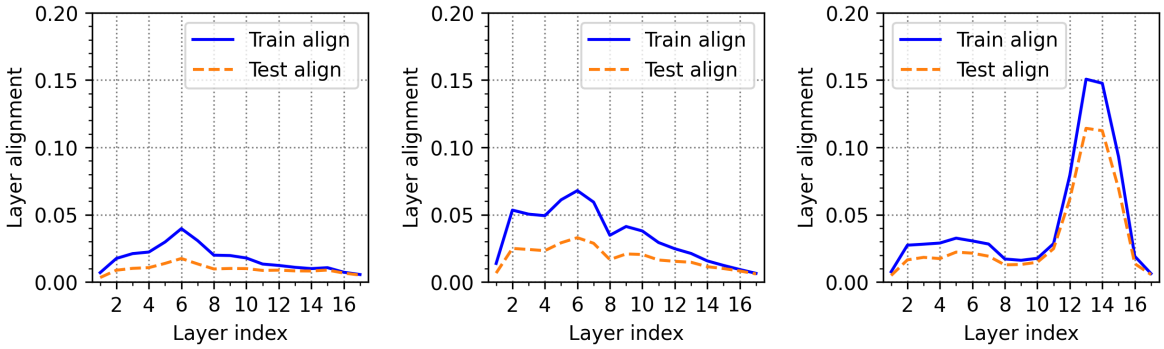


(b) Learning Rate 0.003, test accuracy 56.7%, test loss 1.21



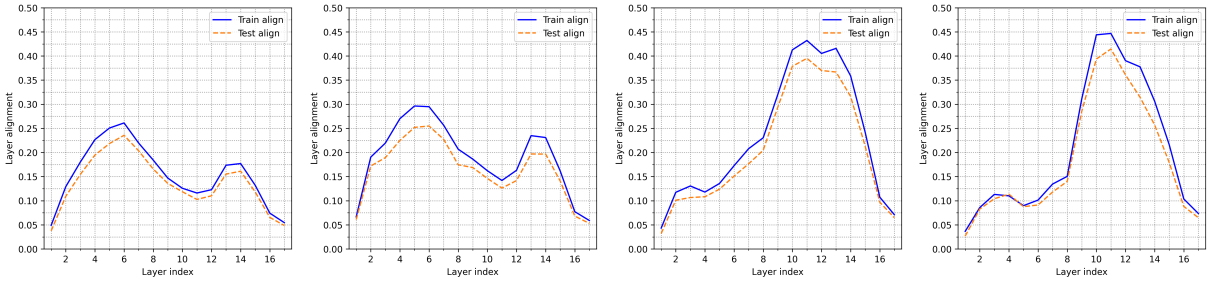
(c) Learning Rate 0.05, test accuracy 51.4%, test loss 1.41

Figure 10: Alignment progress during training. CIFAR10. Further experiments from Fig. 2b.



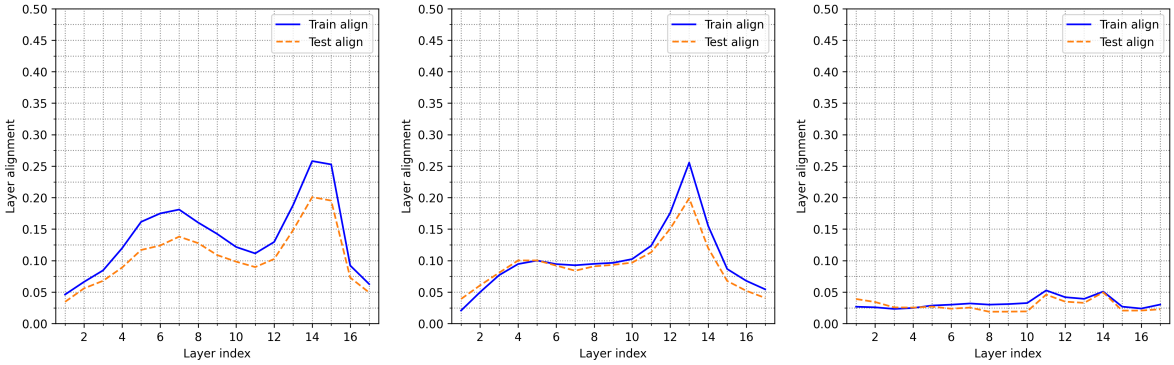
(a) lr 0.001, Acc 46.8% (b) lr 0.005, Acc 53.9% (c) lr 0.01, Acc 62.5%

Figure 11: (a), (b) and (c) show VGG19 trained with SGD (with momentum and weight decay) for 100 epochs on CIFAR100 dataset with three different learning rates (lr). This is an addition to Fig. 4 with more complex architectures. To properly compare the three learning rates, training should be stopped at fixed training loss, as 100 epochs may not allow for convergence with the smaller learning rates (in (a) and (b)).



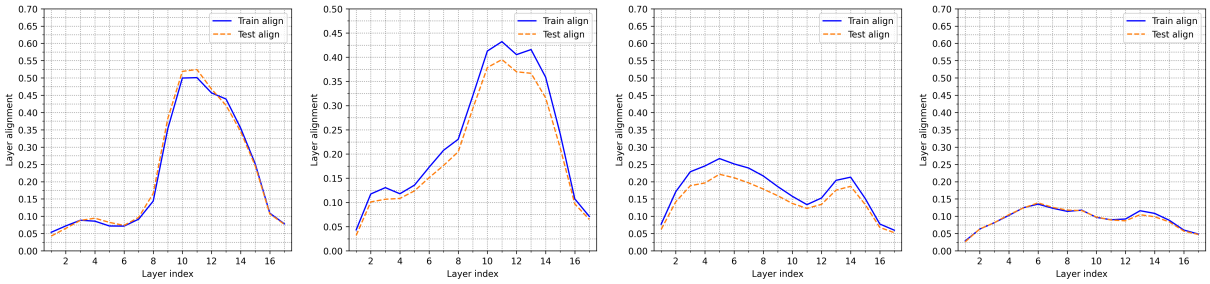
(a) lr 0.0005, Acc Gap 13.1%, Loss Gap 0.560 (b) lr 0.001, Acc Gap 12.3%, Loss Gap 0.484 (c) lr 0.0032, Acc Gap 8.89%, Loss Gap 0.357 (d) lr 0.01, Acc Gap 5.9%, Loss Gap 0.224

Figure 12: (a), (b) and (c) show VGG19 trained with SGD (with momentum and weight decay) until convergence (train loss reaches 0.1) epochs on CIFAR10 dataset with three different learning rates (lr). This is an addition to Fig. 4.



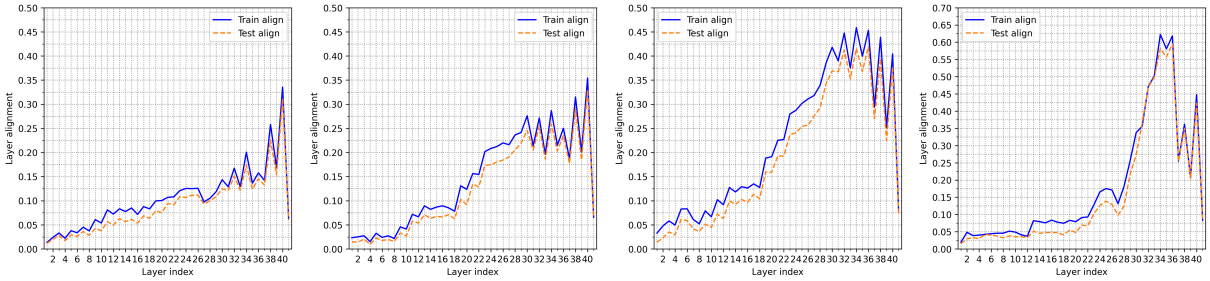
(a) lr 0.000032, Acc Gap 17.4%, Loss Gap 0.769 (b) lr 0.0001, Acc Gap 12.4%, Loss Gap 0.557 (c) lr 0.00032, Acc Gap 11.9%, Loss Gap 0.507

Figure 13: (a), (b) and (c) show VGG19 trained with ADAM until convergence (train loss reaches 0.1) epochs on CIFAR10 dataset with three different learning rates (lr). Note that ADAM exhibits worse gen. error and less salient alignment patterns.



(a) bs 32, Acc Gap 4.7%, Loss Gap 0.236 (b) bs 128, Acc Gap 8.89%, Loss Gap 0.357 (c) bs 512, Acc Gap 11.3%, Loss Gap 0.562 (d) bs 2048, Acc Gap 12.2%, Loss Gap 0.652

Figure 14: (a), (b) and (c) show VGG19 trained with SGD (with momentum and weight decay) until convergence (train loss reaches 0.1) epochs on CIFAR10 dataset with four different batch sizes (bs).



(a) lr 0.001, Acc Gap 10.79%, Loss Gap 0.395 (b) lr 0.003, Acc Gap 9.28%, Loss Gap 0.397 (c) lr 0.01, Acc Gap 4.39%, Loss Gap 0.152 (d) lr 0.02, Acc Gap 5.5%, Loss Gap 0.182

Figure 15: (a), (b) and (c) show ResNet18 trained with SGD (with momentum and weight decay) until convergence (train loss reaches 0.1) epochs on CIFAR10 dataset with four different learning rates (lr).

Details of experiments in Fig. 3: Feed forward Neural Network on CIFAR10, Fashion MNIST and MNIST using SGD optimizer with momentum and weight decay are included in Table 1, Table 2 and Table 3 resp. The learning rates are chosen as the one that produces best out of sample accuracy.

depth	width	learning rate	epochs	test accuracy
10	256	0.005	100	56%
20	256	0.003	100	58.3%
30	256	0.003	150	57.7%
40	256	0.001	200	58.7%
50	256	0.001	250	58%
60	256	0.0007	300	58.3%
70	256	0.0005	300	59.1%
80	256	0.0005	500	57.2%
90	256	0.0001	500	56.9%
100	256	0.0001	700	56%

Table 1: CIFAR10 FFNN experiments to verify EH (Fig. 3c).

depth	width	learning rate	epochs	test accuracy
10	100	0.003	100	88.3%
20	100	0.004	100	88.6%
30	100	0.004	100	89.6%
40	100	0.002	100	88.9%
50	100	0.001	100	88.4%
60	100	0.0007	100	87.7%
70	100	0.0003	200	88.7%
80	100	0.0002	200	87.9%
90	100	0.0001	300	87.6%
100	100	7×10^{-5}	300	88%

Table 2: FMNIST FFNN experiments to verify EH (Fig. 3b).

depth	width	learning rate	epochs	test accuracy
10	100	0.003	100	97%
20	100	0.003	100	97%
30	100	0.003	100	96.9%
40	100	0.002	100	97.6%
50	100	0.001	100	97.6%
60	100	0.0007	100	97.6%
70	100	0.0002	200	94.8%
80	100	0.0001	200	94.4%
90	100	0.0002	300	96.1%
100	100	0.0001	300	95.8%

Table 3: MNIST FFNN experiments to verify EH (Fig. 3a).

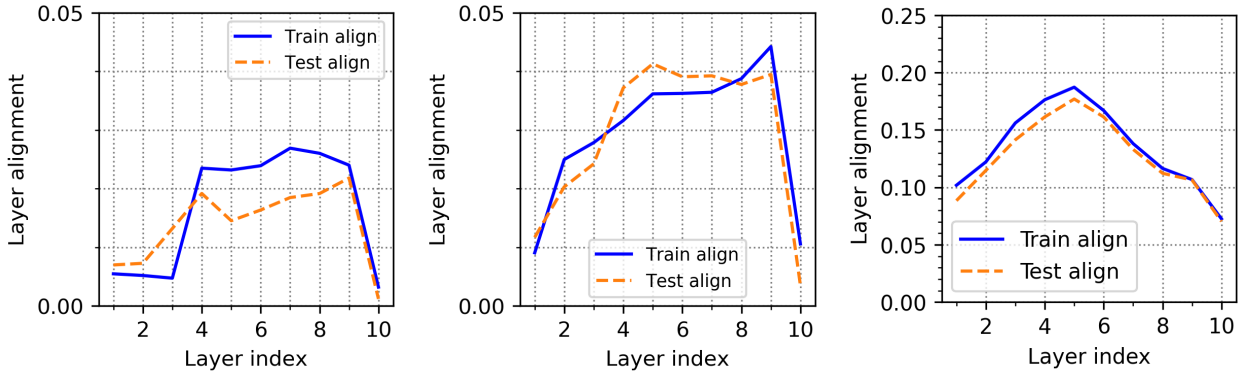
F. Layer-wise alignment of the forward feature kernel

Algorithm 1 Layer-wise maximisation of features

input: DNN N with L layers, LeakyReLU activations ϕ, stochastic optimiser O, batch size b.
input: Training dataset $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ and validation set $V = \{(x_1, y_1), \ldots, (x_{n'}, y_{n'})\}$ with normalised x_i (such that $||x_i||^2 = 1$).
for layers $l = 1, \ldots, L-1$ **do**
  Normalise inputs to l, such that $||\phi(z_{l-1}(x))||^2 = 1$.
  while True **do**
    for Minibatches B in S **do**
      With optimiser O, update weights and biases in layer l using loss function $L(B) = \sum_{x_i, x_j \in B} ||\overrightarrow{K}_l(x_i, x_j) - \frac{1}{2}\delta_{y_i, y_j}||^2$, where $\delta_{y_i, y_j} = 1$ if $y_i = y_j$ else 0, and the unnormalised forward features, $\overrightarrow{K}_l(x_i, x_j) = \phi(z_{l-1}(x_i)) \cdot \phi(z_{l-1}(x_j))$.
    end for
    End while based on increase/plateau of loss on the validation set, $L(V)$.
  end while
  Normalise layer l. Return $\phi(z_l(x))$ as input for layer $l+1$
end for
Train layer L (the final classification layer) with optimiser O and cross entropy loss.
Return N.

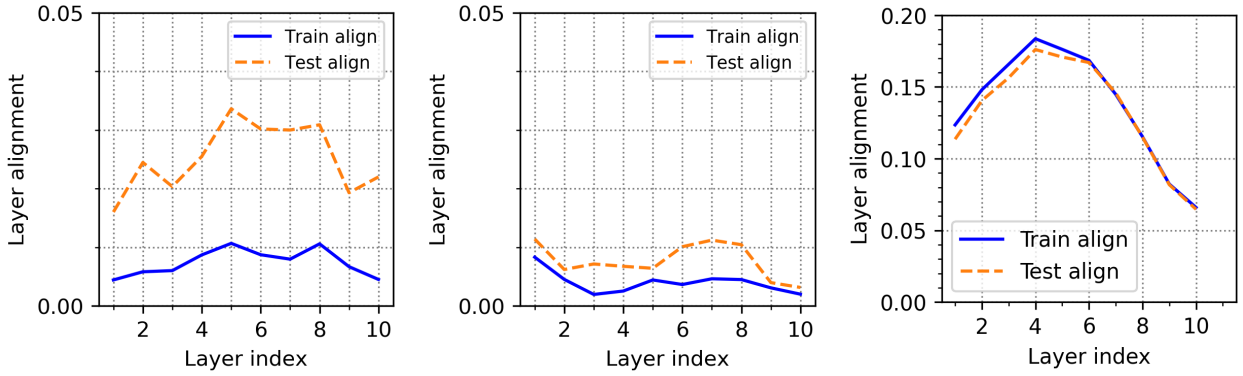
Previous work has studied layer-wise training of neural networks. Here, we adapt a method from Kulkarni and Karande, 2017, which aims to maximise forward feature learning. Here, we confirm that our adapted method generalises well, and show the resulting layerwise CKA on CIFAR10 and MNIST (see Figs. 16 and 17). We will call the algorithm layer-wise

feature maximisation (LFM) – see below. Finally, in Fig. 18, we show the layerwise CKA for networks with all but the last layer frozen during training. We find non-trivial CKA evolution, due to evolution of the backward feature kernel (despite trivial forward features).



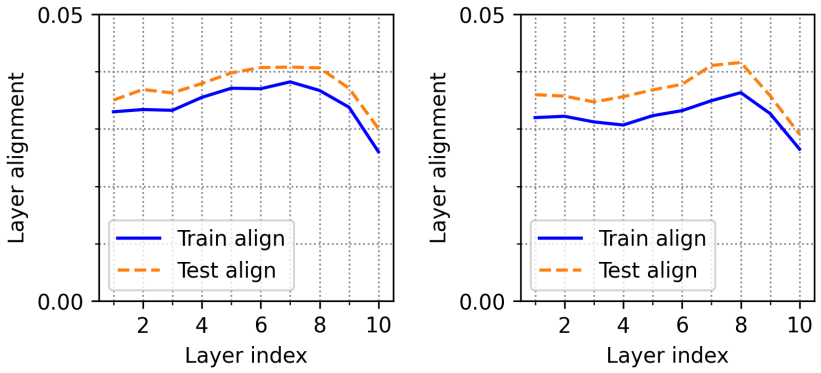
(a) LFM width 32. Acc 93.6% (b) LFM width 256. Acc 95.9% (c) width 32. Acc 96.9%

Figure 16: MNIST. (a) and (b) are trained with LFM (widths 32 and 256 respectively) with ADAM and a learning rate of 0.01, and (c) trained end-to-end with learning rate 0.003. (a) and (b) used a train/validation set split of 45000/5000, and (c) used 50000 training images with no validation set. The LFM does not produce a single peak in the same way an end-to-end trained network does (rather, several layers have approximately maximal alignment). The absolute magnitudes of alignment are also lower for the LFM – determining whether this is an artefact of the layer-wise normalisation scheme or otherwise is a topic of future work. Furthermore, as with experiments on CIFAR10 (Fig. 17), the LFM underperforms relative to the end-to-end neural network. More sophisticated early stopping schemes or different optimisers (Adam was used as other optimisers struggled to converge) may improve generalisation.



(a) LFM width 256. Acc 48.3% (b) LFM width 1024. Acc 49.2% (c) width 256. Acc 56.7%

Figure 17: CIFAR10. (a) and (b) are trained with LFM (widths 256 and 1024 respectively) with ADAM and a learning rate of 0.01, and (c) trained end-to-end with learning rate 0.003. (a) and (b) used a train/validation set split of 45000/5000, and (c) used 50000 training images with no validation set. The alignment for the LFMs is very different to the end-to-end trained system, and the generalisation error is noticeably worse. Understanding why, or improving results is a topic of future work.



(a) Width 2048. Noise scale: 0.01 (b) Width 2048. Noise scale: 1.0

Figure 18: MNIST. All but the last layer is frozen during training (so no feauture learning occurs). In the limit of infinite width, this is equivalent to sampling from an NNGP (A. G. d. G. Matthews et al., 2017). With 10 layers, 2048 is not sufficiently wide to obtain 100% training accuracy (see Table 4), but computing the CKA for each layer scales poorly with layer width. (a) and (b) achieve test accuracies of 91.9% and 88.1% respectively (due to comparatively small layer width). Noise scale determines the scale of the initialisation of the final layer – parameters are sampled i.i.d. from $N(0, s^2 \times 2/L_w)$ (for noise scale s), where L_w is the width of the layer. Decreasing the amount of noise in the last layer appears to shift the peak towards the center, although more careful study is required. Due to the frozen layers, no forward features are learned (so $\overrightarrow{K}_l(x_i, x_j)$ is trivial), but evidently backward features $\overleftarrow{K}_l(x_i, x_j)$ are non trivial. This is unlike neural networks trained end-to-end, which have both non-trivial forward and backward feature kernels.

Dataset	Number of layers	Width	Max train acc	Max test acc
MNIST	10	75000	97.6%	96.4%
MNIST	10	10000	96.4%	95.6%
MNIST	10	2048	92.6%	91.9%

Table 4: Best train/test accuracy for frozen neural networks as a function of layer widths. Network parameters are initialised i.i.d. from $N(0, 2/L_w)$ where L_w is the width of the layer. Clearly, layers have to be very wide before near parity can be achieved with finite width unfrozen networks.

G. Connecting CKA with Effective Rank, NTK, and Fisher information

In the following appendix, we will summarize work related to the effective rank of Ψ and their connection to CKA A, and provide some related novel experiments. Baratin et al., 2021; Oymak et al., 2019 independently observed that the following phenomenon holds when DNNs generalize:

1. Ψ has a small number of large singular values while most other singular values are much smaller.

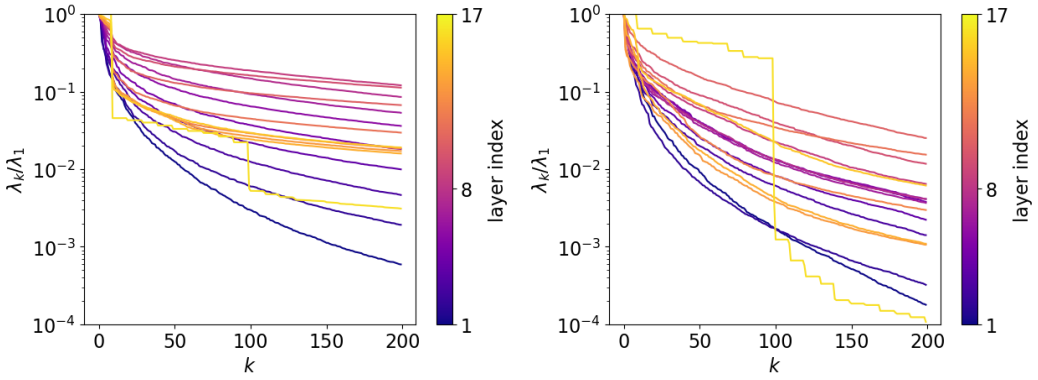
2. The label vector Y is aligned with large singular directions in Ψ.

The two conditions above are also the conditions for maximizing CKA. As was argued in Baratin et al., 2021, the CKA was introduced as the measure of model compression and feature selection. The first condition can be viewed as the measure of model compression, as effectively only a few directions in parameter space are relevant for changes of the function. The second condition can be interpreted as feature selection because we want the anisotropy of the tangent space to be skewed toward directions that leads to correct labels in function space.

We will first show that above observations still hold for layerwise CKA in Appendix G.1. Then, in Appendix G.4, we shed more light into the connection between the two conditions and the generalization via the Fisher information matrix.

G.1. Connection between kernel alignment A and the Effective Rank of $\Psi^T\Psi$

Let us begin with observation 1: that only a handful of eigenvalues are large. As shown in Fig. 19, the eigenvalues differ in logarithmic scale for both functions that do and do not generalize. Even though both eigenvalue distributions are spread on a logarithmic scale, the magnitude of the spread is different. To quantify this, we introduce the stable rank (Rudelson and Vershynin, 2007).[11]



(a) Trained on 20% random labels, test accuracy = 70.41%

(b) Trained without random labels, test accuracy = 90.59%

Figure 19: The top 200 normalized eigenvalues of $C{\Psi_1}^T\Psi_1 C$ for different layers l. VGG19 was trained on CIFAR10 (a) with 20% random labels and (b) without random labels. The relative eigenvalues decay on logarithmic scale for both cases. The logarithmic gap in eigenvalues assert that $C{\Psi_1}^T\Psi_1 C$ has low effectively rank. On average, the eigenvalue drops faster for generalizing case (b) compared to less generalizing case (a).

For a matrix W of rank k, the stable rank is

$$R(W) = \frac{||W||_F^2}{||W||_2^2} = \frac{\sum_i \lambda_i^2}{\lambda_1^2},$$

where the numerator and denominator are squares of the Frobenius norm and spectral norm respectively. Singular values λ_i are given in descending orders measured by their absolute values. The stable rank is scale invariant and upper bounded by the true rank. Note that stable rank becomes 1 when $\lambda_1 >> \lambda_j$ for all $j > 1$.

Using the definition of stable rank, layerwise CKA can be written as

$$\begin{aligned} A_l &= \frac{\tilde{Y}^T C{\Psi_1}^T\Psi_1 C\tilde{Y}}{||C{\Psi_1}^T\Psi_1 C||_F} = \frac{\lambda_1}{||C{\Psi_1}^T\Psi_1 C||_F}\frac{\tilde{Y}^T C{\Psi_1}^T\Psi_1 C\tilde{Y}}{\lambda_1} \\ &= \frac{1}{\sqrt{R_l}}\left(\sum_i \frac{\lambda_i}{\lambda_1}\left\langle u_i, \tilde{Y}\right\rangle^2\right), \end{aligned}$$

where R_l is the stable rank of $C{\Psi_1}^T\Psi_1 C$ and λ_i, u_i are its eigenvalue and eigenvector respectively. We have assumed $\tilde{Y}$ has been normalized. Then, layerwise CKA can be divided into two terms: inverse square root of stable rank of $C{\Psi_1}^T\Psi_1 C$ and the weighted correlation between the u_i and $\tilde{Y}$. Each term is related to the observation 1 and 2 respectively.

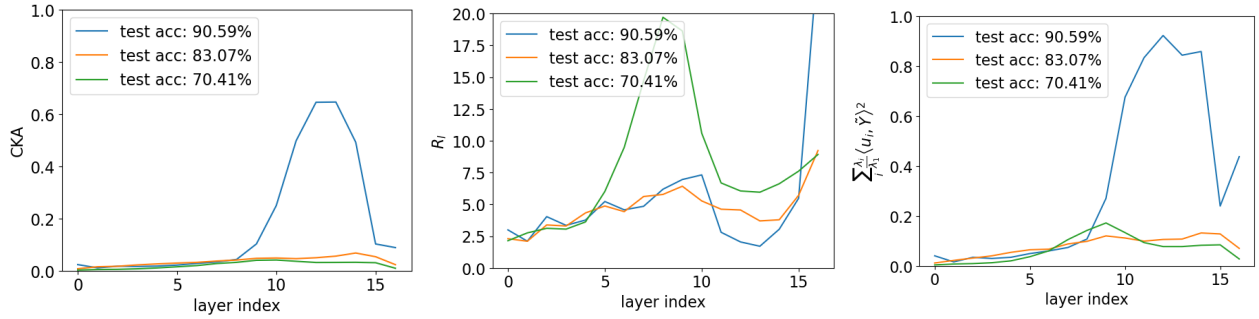
Notice that A_l is maximized when R_l is minimized and the correlation term $\left(\sum_i \lambda_i/\lambda_1 \left\langle u_i, \tilde{Y}\right\rangle^2\right)$ is maximized. This result is trivial because CKA is maximizing the alignment with $\tilde{Y}\tilde{Y}^T$, which is a rank 1 matrix with $\tilde{Y}$ as its eigenvector.

We observe in Fig. 20b that the stable rank is consistently small (i.e. concentrated eigenvalues) for most of the layers (this relates to Observation 1). In addition, the stable rank is smaller when better generalization gap is achieved. This could be

[11] Other measures of effective rank (e.g. Roy and Vetterli, 2007) are also valid for our study as long as it can represent the exponential gap in eigenvalues.

interpreted as signifying that only a few directions in parameter space can meaningfully alter the function, hence a function is more robust against perturbation in parameters, and effectively lower dimensional.

G.2. Connection between kernel alignment A and the Correlation $\sum_i \lambda_i/\lambda_1 \left\langle u_i, \tilde{Y} \right\rangle^2$



(a) CKA (b) Stable rank (c) Correlation

Figure 20: Plots for (a) CKA A_l, (b) stable rank R_l, and (c) correlation $\sum_i \frac{\lambda_i}{\lambda_1} \left\langle u_i, \tilde{Y} \right\rangle^2$ after training. VGG19 was trained on CIFAR10 with following conditions to achieve different test accuracy: (blue) 90.59%, lr=0.02; (orange) 83.07%, lr=0.001; (green) lr=0.001 and 20% random labels. (a) Higher test accuracy correlates with larger layer-wise CKA, as suggested by Fig. 4 in the main text. (b) The stable rank is on average larger for a model trained with the random labels (green), while smaller for models trained without the random labels (blue, orange). This suggests that low rank is the first condition that must be satisfied for good generalization. The importance of low rank can be seen from the fact that it implies robustness to perturbation in parameters. (c) The correlation is large only for best generalising model (blue), and is small otherwise. This suggests that large correlation may be a second condition for generalization. A large correlation indicates that the only allowed deviation in function space is along $\tilde{Y}$.

As mentioned before, the correlation term measures how much $\tilde{Y}$ is aligned with the large eigenspace of $C{\mathbf{\Psi_1}}^T\mathbf{\Psi_1}C$: the quantitative measurement for observation 2. The correlation must also be large in order for A_l to be large. When such is the case, we may approximate

$$\Psi_l \approx v_l \sqrt{\lambda_1} \tilde{Y}^T, \tag{26}$$

and thus the effect of perturbation in θ on f,

$$\Delta f^T \approx \langle \Delta\theta, v_l \rangle \sqrt{\lambda_1} \tilde{Y}^T. \tag{27}$$

Thus, any infinitesimal change in parameter space for most directions cannot alter f. The only allowed direction of change is along $\tilde{Y}$. The direction along $\tilde{Y}$ is special because f cannot be effectively changed to increase the probability of an incorrect label. This combined with observation 1 suggests that most directions in parameter space are robust against perturbation, and the only meaningful direction of change is along the direction that uniformly increases the function values of misclassified labels.

The correlation term for models of different generalization can be seen from Fig. 20c. Notice that the correlation is small for all of the layers for the two worse performing models (orange, green). However, for the model with the best generalization error (blue), the correlation increases to near 1 in the intermediate layers. The reason for this alignment hierarchy (due to the training process) was explained in Section 3 of the main text.

G.3. Fisher information matrix

Fisher information is the metric tensor of a statistical manifold (Amari and Nagaoka, 2000), and it provides the local measure of how fast a prediction of model changes according to change in parameters. Because DNNs are commonly trained using the gradients of the negative log likelihood (NLL), and not the gradient of the functions f_θ[12], Fisher information becomes a

[12]For MSE loss, the two become the same

natural choice for calculating the flatness of different hypotheses. Fisher information $I^{exp}(\theta)$ is

$$I_{ij}^{exp}(\theta) = \mathbf{E}_{x\sim q(x)}\left[\mathbf{E}_{y\sim p(y|x;\theta)}\left[\frac{\partial \log(p(y|x;\theta))}{\partial\theta_i}\frac{\partial \log(p(y|x;\theta))}{\partial\theta_j}\bigg|_\theta\right]\right],$$

where $q(x)$ is the true distribution of the inputs, and $p(y|x;\theta)$ is the conditional probability predicted by a statistical model at θ. To see why it is a measure of sensitivity of the likelihood along the parameters, Fisher information can be equivalently expressed as

$$I_{ij}^{exp}(\theta) = -\mathbf{E}_{x\sim q(x)}\left[\mathbf{E}_{y\sim p(y|x;\theta)}\left[\frac{\partial^2 \log(p(y|x;\theta))}{\partial\theta_i\partial\theta_j}\right]\right] = \frac{\partial^2 D_{KL}\left(p(y|x;\theta)||p(y|x;\theta+\Delta\theta)\right)}{\partial\theta_i\partial\theta_j}\bigg|_{\Delta\theta=0},$$

which is the curvature of KL divergence at θ. Thus, quantifying the second order changes of $p(y|x;\theta)$ (i.e. large eigen direction of $I_{ij}^{exp}(\theta)$ leads to sharper change of $p(y|x;\theta)$, while small eigen direction leads to flatter changes). This quantity is more useful near the maximum likelihood estimate when the first order deviation terms disappear.

For our purposes, we will only consider the empirical fisher information

$$I(\theta) = \mathbf{\Psi}\frac{\partial^2\mathcal{L}}{\partial F^2}\mathbf{\Psi}^T,$$

where $\mathcal{L}$ is the negative log likelihood, and $F \in \mathbb{R}^{o\times n}$ is the concatenated network output. We have dropped the centering matrix C following the observation from Baratin et al., 2021 that quantitatively similar results were obtained for CKA and KA. For the special case when $\mathcal{L}$ is MSE loss[13],

$$\frac{\partial^2\mathcal{L}}{\partial F_\alpha \partial F_\beta} = \frac{\partial^2 \sum_i^{on}(F_i - y_i)^2}{\partial F_\alpha \partial F_\beta} = \delta_{\alpha\beta},$$

and $I(\theta) = \mathbf{\Psi}\mathbf{\Psi}^\mathbf{T}$. Therefore, $I(\theta)$ and NTK share same set of non-zero eigenvalues for MSE loss. For more general discussion beyond the empirical case, see Appendix A of Baratin et al., 2021.

G.4. Fisher Information and Generalization via the Stable Rank

It has been argued that "flatness" (albeit not being the sole factor) is related to good generalization (Keskar et al., 2016; Wu, Zhu, et al., 2017; Neyshabur et al., 2017). The flatness is calculated by the Hessian, which is equal to the Fisher information for MLE. Thus, measuring the stable rank of Fisher information can be a measure of the flatness of the model. Flatness of f may be inadequate, because sharpness on f does not always lead to sharpness on $\mathcal{L}$ (e.g. when the softmax function is saturated).

We will explore the case of cross-entropy loss to see what CKA can infer about the flatness of the loss landscape. For such settings,

$$\frac{\partial^2\mathcal{L}}{\partial F\partial F} = \begin{pmatrix} \frac{\partial^2\mathcal{L}}{\partial f(x_1)\partial f(x_1)} & & & \\ & \frac{\partial^2\mathcal{L}}{\partial f(x_2)\partial f(x_2)} & & \\ & & \ddots & \\ & & & \frac{\partial^2\mathcal{L}}{\partial f(x_n)\partial f(x_n)} \\ , & & & \end{pmatrix}$$

where

$$\frac{\partial^2\mathcal{L}}{\partial f_\theta^i(x)\partial f_\theta^j(x)} = \delta_{ij}p_i(x) - p_i(x)p_j(x),$$

and $p_i(x)$ is

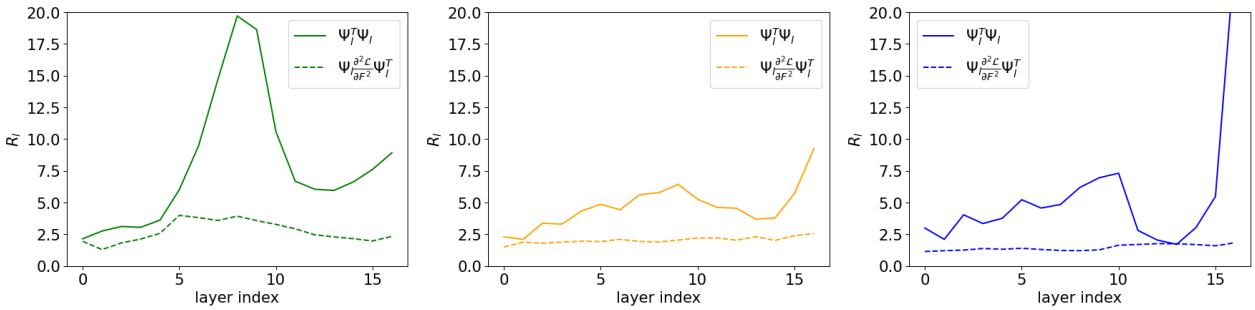
$$p_i(x) = p(y=i|x;\theta) = \frac{e^{f_\theta^i(x)}}{\sum_j e^{f_\theta^j(x)}}.$$

[13] MSE loss is used for NTK, and can be interpreted as the likelihood being a Gaussian distribution with fixed covariance.

$\frac{\partial^2 \mathcal{L}}{\partial F \partial F}$ is clearly different from the identity matrix and approaches 0 as the training converges. However, it is still possible to infer the properties of Fisher information from $\mathbf{\Psi}$.

By the property of rank under matrix multiplication, $\text{rank}(\mathbf{\Psi} \frac{\partial^2 \mathcal{L}}{\partial F^2} \mathbf{\Psi}^T) \leq \text{rank}(\mathbf{\Psi^T \Psi})$. Even though the stable rank does not strictly satisfy this condition, we can infer that the stable rank of Fisher information would still be similar or smaller than that of $\mathbf{\Psi^T \Psi}$. In Fig. 21, the rank of Fisher information indeed is observed to be upper bounded by the rank of $\mathbf{\Psi^T \Psi}$. More surprisingly, the stable rank of Fisher information is consistently small across all layers, which is left as a future direction of investigation.

As seen in Fig. 22, the stable rank of Fisher information not only is upper bounded by the stable rank of CKA, but loosely follows the trend of CKA. In addition, generalizing models leads to smaller stable rank of Fisher information. We can empirically postulate that larger CKA leads to smaller stable rank of CKA, which in turn informs us about the stable rank of Fisher information.

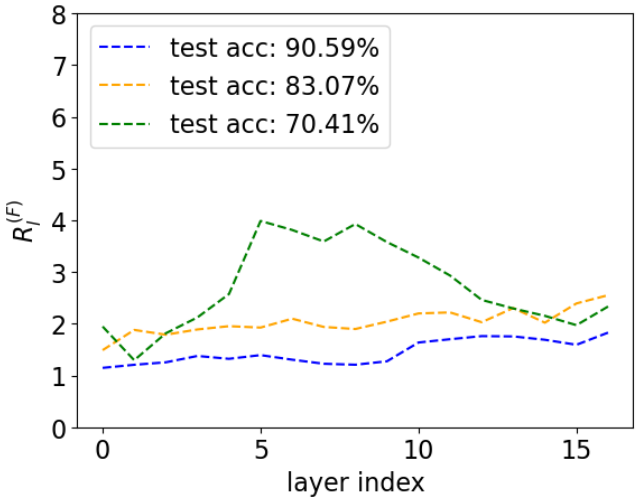


(a) 20% random labels, test accuracy = 70.41%

(b) no random labels, test accuracy = 83.07%

(c) no random labels, test accuracy = 90.59%

Figure 21: Stable rank R_l for NTK ($\mathbf{\Psi^T \Psi}$) (solid) and Fisher information rank($\mathbf{\Psi} \frac{\partial^2 \mathcal{L}}{\partial F^2} \mathbf{\Psi}^T$) (dashed) for models trained to different test accuracies. The training condition is equal to that of Fig. 20, denoted by the same colors. Even though inequality of ranks for matrix multiplication does not hold strictly, it is observed that the rank of NTK upper bounds that of Fisher information. In addition, the stable rank of Fisher information is more consistent over the layers.



(a) Ranks of Fisher information

Figure 22: The comparison of the ranks of Fisher information. Higher test accuracy correlates with smaller stable rank. It can be inferred from the fact that lower stable rank leads to more robustness in the prediction probabilities.

G.5. Fisher Information and Correlation

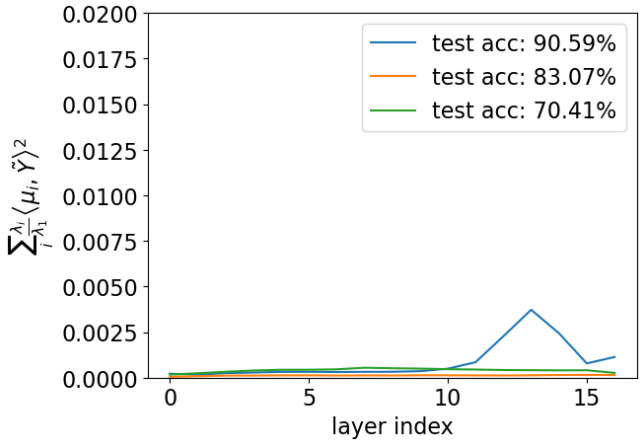
Even though the empirical relationship between the stable rank of Fisher information and CKA is clear, the correlation term is less straightforward. For the CKA, the correlation is defined between the anisotropy of the tangent space with our direction of interest ($\tilde{Y}$) in function space. We can extend it to measurement of the anisotropy of tangent space introduced by the Fisher information. First, let us define a square root of Fisher information as

$$\sqrt{I_l} = \sqrt{\frac{\partial^2 \mathcal{L}}{\partial F \partial F}} \mathbf{\Psi_1} = \sum_i \mu_i^T \sqrt{\lambda_i^{(F)}} \nu_i,$$

where μ_i and ν_j are left and right singular vectors respectively, and $\lambda_i^{(F)}$ is the corresponding eigenvalue of Fisher information. Then we define the Fisher correlation as

$$\sum_i \frac{\lambda_i^{(F)}}{\lambda_1^{(F)}} \left\langle \mu_i, \tilde{Y} \right\rangle^2.$$

However, as seen in figure Fig. 23, the correlation is not evident from the experiments. The investigation of why the correlation disappears for Fisher information is left as future work.



(a) Correlation for Fisher information

Figure 23: The comparison of the correlation of Fisher information for models with different accuracies. The experiment conditionas are equal to that of Fig. 20. The correlation is too small for meaningful argument.