Reactivation: Empirical NTK Dynamics Under Task Shifts

author names withheld

Under Review for the Workshop on High-dimensional Learning Dynamics, 2025

Abstract

The Neural Tangent Kernel (NTK) offers a powerful tool to study the functional dynamics of neural networks. In the so-called lazy, or kernel regime, the NTK remains static during training and the network function is linear in the static neural tangents feature space. The evolution of the NTK during training is necessary for feature learning, a key ingredient of the success of deep learning. The study of the dynamics of the NTK has led to several critical discoveries in recent years, in generalization and scaling behaviours. However, this body of work has been limited to the single task setting, where the data distribution is assumed constant over time. In this work, we present a comprehensive empirical analysis of NTK dynamics in continual learning, where the data distribution shifts over time. Our findings highlight continual learning as a rich and underutilized testbed for probing the dynamics of neural training. At the same time, they challenge the validity of static-kernel approximations in theoretical treatments of continual learning, even at large scale.

1. Introduction

Continual learning is central to real-world applications where models must adapt to a sequence of tasks without forgetting previous ones. While architectural and algorithmic advances have been proposed to tackle this problem, its underlying learning dynamics remain underexplored.

Recent advances in deep learning theory, particularly the introduction of the Neural Tangent Kernel (NTK) framework [7], offer a powerful lens for analyzing training behavior. Jacot et al. [7] showed that infinitely wide neural networks evolve like kernel machines, with learning dynamics governed by a kernel matrix fixed at initialization. However, in most practical settings the NTK evolves during training, allowing for features to adapt to the data distribution. The NTK framework gave rise to a dichotomy between two distinct training regimes: the **lazy (or kernel) regime**, where the network's internal representations remain largely fixed and learning occurs primarily through adjustments to final-layer weights; the **rich (or feature learning) regime**, where the network's representations evolve substantially during training, allowing more expressive modeling.

Fort et al. [6] analyzed the empirical evolution of the NTK across width and depth and found that in practical networks, the NTK changes substantially during training, correlating with improved performance and stronger feature learning. Other studies have focused on the structure of kernel evolution. A prominent phenomenon is the *kernel alignment*: the tendency for NTK eigenvectors to align with task-relevant directions over time, enhancing generalization and learning efficiency [1, 12]. In parallel, works on loss landscape geometry have revealed additional dynamics linked to NTK behavior. The phenomenon of *progressive sharpening* —an increase in the curvature of the loss landscape during early training—has been observed in both deep and wide networks [3, 8]. These sharpness dynamics correspond with periods of high NTK change due to the relation between the NTK and Hessian spectra.

These analyses rely on the assumption of *stationarity*—that training data is sampled from a fixed distribution throughout. This assumption breaks down in continual learning settings, where tasks arrive sequentially and data distributions shift over time. This raises a fundamental and largely open question:

> *How do learning dynamics—particularly those captured by the NTK—respond to distributional shifts across tasks?*

Some recent theoretical efforts have developed models of *continual learning performance* in the lazy regime. For example, Bennani et al. [2], Doan et al. [5], Karakida and Akaho [9] study learning curves in continual settings within a static-kernel approximation. While analytically tractable, such models implicitly assume that NTKs remain nearly constant—even in finite-width networks—thus failing to capture the full richness of learning dynamics observed in practice. This disconnect highlights a critical gap between theory and practice in continual learning.

1.1. Contributions

This work provides a systematic, empirical investigation of Neural Tangent Kernel (NTK) dynamics in the context of continual learning—a setting that challenges the conventional assumption of stationary data distributions. Our contributions are as follows:

1. We evaluate how NTK dynamics respond to changes in network width, learning rate, training duration, and—critically—task similarity, across single and multiple task switches.
2. We demonstrate that task transitions consistently trigger abrupt shifts in the NTK, even in wide networks typically associated with lazy learning, revealing a reactivation of feature dynamics at each task boundary.
3. Through controlled experiments, we distinguish between different types of distributional shifts, showing that the introduction of semantically novel classes leads to significantly greater NTK change.

By systematically characterizing how NTKs evolve in non-stationary regimes, our results highlight continual learning as a promising and underexplored testbed for studying training dynamics.

2. Experiments and Results

2.1. NTK Metrics

We review the definition and some fundamental ideas related to the Neural Tangent Kernel in Appendix A. Here, we introduce the main metrics used in our experiments. The experiments presented consist in image classification tasks on CIFAR and ImageNet for several seeds. More training details can be found in Appendix B.

Kernel Spectral Norm It is equivalent to the max eigenvalue of NTK. We show in Appendix A.1 that the NTK spectral norm controls the convergence rate in certain eigenmodes.

Kernel Distance In line with Fort et al. [6], we define the kernel distance based on Centered Kernel Alignment $\mathrm{CKA}(\cdot,\cdot)$ ([4],[10], see definition in the Appendix A) as:

$$S(\Theta,\Theta') \triangleq 1-\mathrm{CKA}(\Theta,\Theta') \tag{1}$$

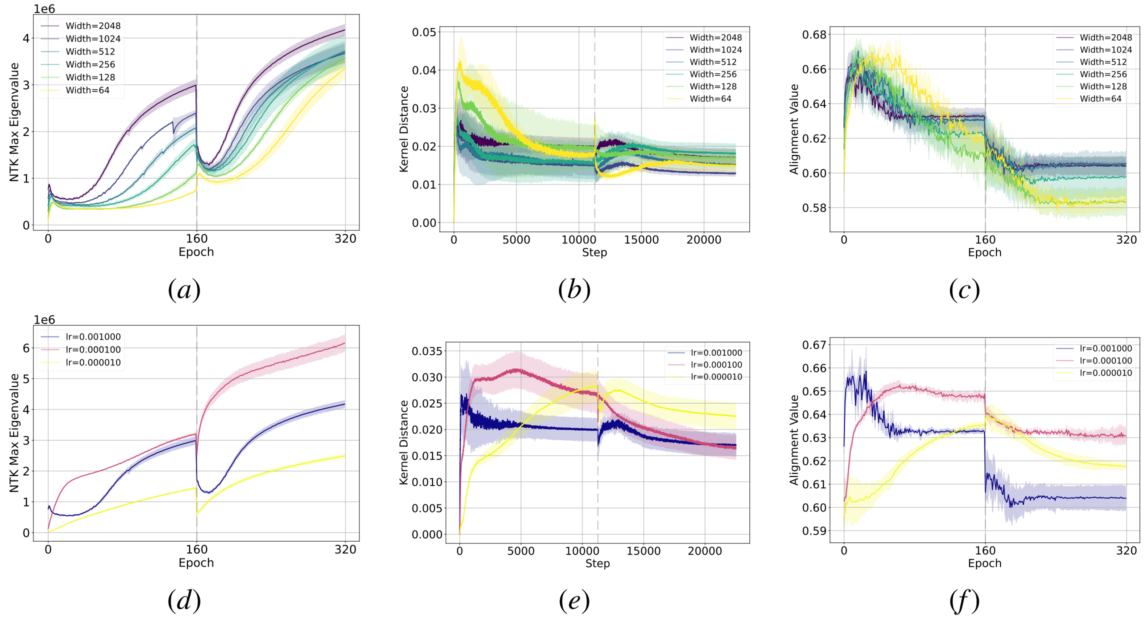
Kernel Velocity The kernel velocity $v(t)$ quantifies the rate of change of NTKs at time t:

$$v(t) \triangleq S\big(\Theta_t, \Theta_{t+dt}\big)/dt \tag{2}$$

Kernel Alignment The kernel alignment $A(t)$[4] at time t measures the similarity between the NTK and the target label kernel $\mathbf{y}\mathbf{y}^\top$ (where $\mathbf{y}$ is the label vector):

$$A(t) \triangleq \mathrm{CKA}(\Theta_t, \mathbf{y}\mathbf{y}^\top) \tag{3}$$

2.2. Task Shifts Reactivate Learning Dynamics



(*a*) (*b*) (*c*)

(*d*) (*e*) (*f*)

Figure 1: Comparison of NTK Max Eigenvalue (Column 1), Kernel Distance (Column 2), and Kernel Alignment (Column 3) across different widths (Row 1: fix lr=0.0001) and learning rates (Row 2: fix width=2048) during a single task switch with 5 classes in each task.

Scaling up neural networks—along with appropriate learning rate rescaling [13]—is known to induce the lazy regime, in which training occurs in a nearly linear function space and the network's internal features remain effectively static. This regime has become especially attractive in continual learning due to its analytical tractability. In our experiments, we confirm this trend: as model width increases, NTK dynamics become increasingly lazy, as indicated by reduced kernel distance and velocity during training (Figure 1b; Appendix Figure 9). However, at the moment of task switch, we observe a clear and consistent spike in kernel velocity, signaling a temporary departure from the lazy regime. The network briefly enters a dynamic phase of feature adaptation before quickly returning to stability. We refer to this phenomenon as the **re-activation** of feature learning.

This reactivation is accompanied by a sharp drop in the NTK norm at the onset of the new task, followed by a gradual recovery. This creates a distinctive asymmetric V-shape or "check-mark" trajectory in the NTK norm—observed consistently across all model widths. The timing of this drop aligns with the spike in velocity, suggesting a rapid reconfiguration of the network's functional representation in response to the new task. Similar patterns are observed in Kernel Alignment (Fig1c,f), indicating that the NTK rapidly changes direction at the task switch and begins evolving along a new direction. This directional shift is much more slower for *under-converged* phase. We also find that these patterns persist across multiple task switches, as shown in Figure 7 in the appendix.

We find that the behavior at task switch critically depends on the degree to which the network has been trained. By changing the learning rate, we approximately categorize training phases at the task switch as, respectively, *under-converged*, *converged*, and *over-converged*, corresponding to $lr = \{0.00001, 0.0001, 0.001\}$, based on the accuracy curves shown in Figure 5a.

Notably, the speed of the drop in NTK norm varies among configurations: in the over-converged phase the network takes the longest time to recover from the disruption introduced by the task switch.

2.3. Task Similarity Controls NTK Dynamics

Section 2.2 described the reactivation phenomenon in learning dynamics during task switches. In this section we dissect the phenomenon further, looking into the nature of the task switch. A task switch introduces a shift in the data distribution, which cause the reactivation of feature learning. However, ”distribution shift” is a generic term which can be mapped to many different scenarios. In particular, we consider two specific cases of distribution shift in this work: the introduction of new classes, and the change of the relative frequencies of a set of known classes.

In the first case, for each experiment E_i, the network is trained on distribution $\mathcal{D}_0$ in task 1 and $\mathcal{D}_i$ in task 2, where $\mathcal{D}_k$ denotes a uniform mixture over 10 classes $\{k, k+1, \dots, k+9\}$. Thus the similarity between $\mathcal{D}_0$ and $\mathcal{D}_i$ can be measured as the overlap between the classes:

$$\text{Similarity}(\mathcal{D}_0, \mathcal{D}_i) = \frac{|\mathcal{D}_0 \cap \mathcal{D}_i|}{|\mathcal{D}_0 \cup \mathcal{D}_i|}.$$

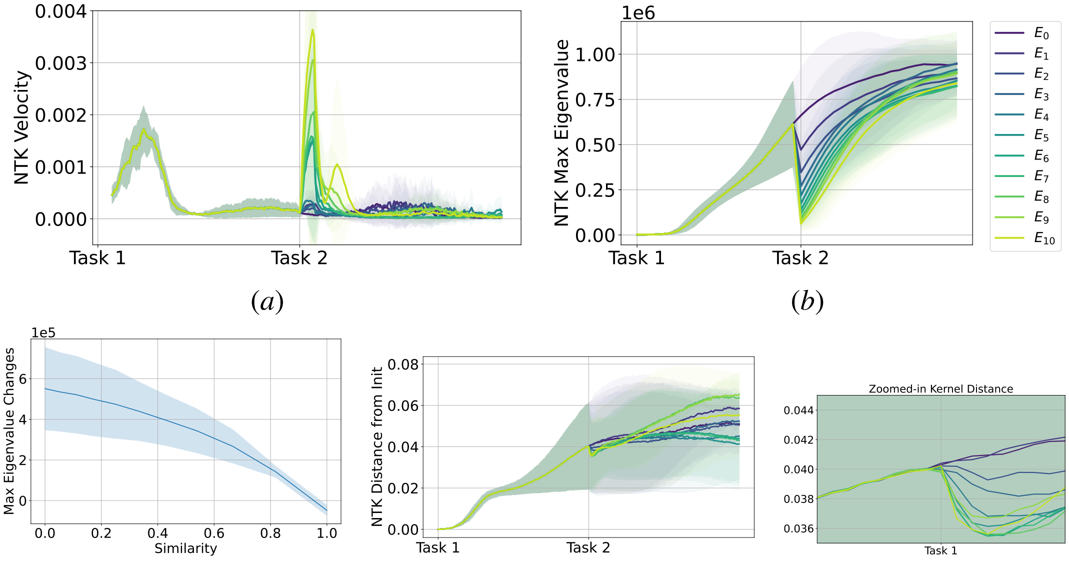
By varying $i \in [0, 1]$, we sweep the similarity between 1 (identical tasks) and 0 (no class overlap, a typical benchmark for continual learning).

In the second case, we define two disjoint class subsets $\tilde{\mathcal{D}}_0$ and $\tilde{\mathcal{D}}_1$, and interpolate between them with mixtures: $\tilde{\mathcal{D}}_\alpha = (1-\alpha)\tilde{\mathcal{D}}_0 + \alpha\tilde{\mathcal{D}}_1$. For each experiment $\tilde{E}_\alpha$ the network is trained on $\tilde{\mathcal{D}}_{0.1}$ in task 1 and $\tilde{\mathcal{D}}_\alpha$ in task 2, varying $\alpha \in [0.1, 0.9]$. The similarity metric is linear:

$$\text{Similarity}(\tilde{\mathcal{D}}_\alpha, \tilde{\mathcal{D}}_\beta) = 1 - |\alpha - \beta|.$$

When the new task introduces new concepts (experiments $E_0 \dots, E_{10}$), there is a direct relationship between the number of new classes and the amount of change in the NTK, as confirmed in the measurement of NTK norm, velocity and kernel distance (Figure 2). We observe the characteristic check-mark shape in all but the E_0 case, where no distributional change occurs (Figure 2a-c). The drop in NTK norm becomes progressively smaller as class overlap increases, revealing a clear monotonic relationship between task similarity and the magnitude of NTK disruption.

Again, the NTK norm recovers gradually after the drop, consistent across all levels of similarity (Figure 2b). A similar trend is observed in the kernel distance (Figure 2d-e), where larger distribution shifts cause more pronounced deviations from the previous NTK state. The trend is neatly ordered by task similarity, suggesting the existence of an underlying law governing the NTK spectral evolution, parametrized by the task similarity. Further, the kernel velocity (Figure 2a) confirms that most of the feature learning occurs immediately following the task switch, after which the network appears to settle back into a more stable regime within a few epochs.



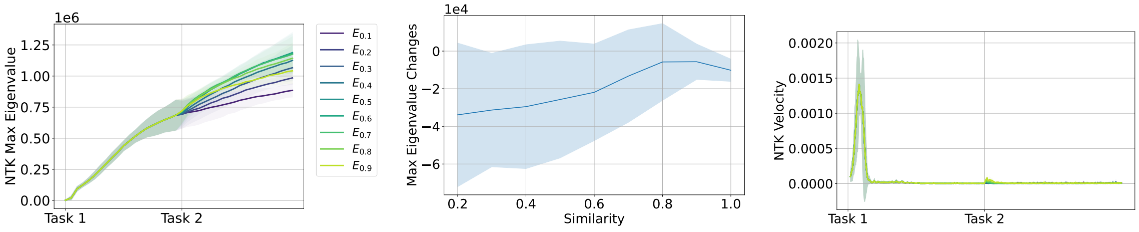
(*a*) (*b*)

(*c*) (*d*) (*e*)

Figure 2: NTK Dynamics with concept-based distribution shift.

We also note a diminishing return effect: the introduction of the first few new classes causes disproportionately large changes in NTK, while later additions have more incremental impact—suggesting a sublinear relationship between the number of new concepts and NTK disruption.

The picture is very different if the new task does not introduce new classes, as in the experiments E_α with $\alpha \in [0, 1]$. Figure 3 shows that NTK changes in this case are significantly smaller than in Experiment 1. The NTK norm (Figure 3a) evolves smoothly without any sharp discontinuity at the task switch. Likewise, the kernel velocity (Figure 3c) remains low, indicating that feature reactivation does not occur in response to proportion shifts alone. Although some monotonic trends are still visible in the NTK eigenvalues (Figure 3b), their scale is minor.



(*a*) NTK Max Eigenvalue (*b*) Max Eigenvalue Changes (*c*) Kernel Velocity

Figure 3: NTK Dynamics with frequency-based distribution shift.

3. Conclusion

This work unveils the learning dynamics in continual learning through the lens of empirical NTK evolution. Our findings are consistent with prior observations in stationary settings and further reveal that distribution shifts across tasks induce significant changes in NTK behavior. These results highlight the transition from lazy to feature learning as new tasks arrive, and we hope they inspire further theoretical and empirical investigation into learning dynamics in continual learning.

References

[1] Aristide Baratin, Thomas George, César Laurent, R Devon Hjelm, Guillaume Lajoie, Pascal Vincent, and Simon Lacoste-Julien. Implicit regularization via neural feature alignment. In *International Conference on Artificial Intelligence and Statistics*, pages 2269–2277. PMLR, 2021.

[2] Mehdi Abbana Bennani, Thang Doan, and Masashi Sugiyama. Generalisation guarantees for continual learning with orthogonal gradient descent. *arXiv preprint arXiv:2006.11942*, 2020.

[3] Jeremy Cohen, Simran Kaur, Yuanzhi Li, J Zico Kolter, and Ameet Talwalkar. Gradient descent on neural networks typically occurs at the edge of stability. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=jh-rTtvkGeM.

[4] Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. Algorithms for learning kernels based on centered alignment. *The Journal of Machine Learning Research*, 13(1):795–828, 2012.

[5] Thang Doan, Mehdi Bennani, Bogdan Mazoure, Guillaume Rabusseau, and Pierre Alquier. A theoretical analysis of catastrophic forgetting through the ntk overlap matrix, 2021. URL https://arxiv.org/abs/2010.04003.

[6] Stanislav Fort, Gintare Karolina Dziugaite, Mansheej Paul, Sepideh Kharaghani, Daniel M Roy, and Surya Ganguli. Deep learning versus kernel learning: an empirical study of loss landscape geometry and the time evolution of the neural tangent kernel. *Advances in Neural Information Processing Systems*, 33:5850–5861, 2020.

[7] Arthur Jacot, Franck Gabriel, and Clement Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/5a4be1fa34e62bb8a6ec6b91d2462f5a-Paper.pdf.

[8] Stanislaw Jastrzebski, Maciej Szymczak, Stanislav Fort, Devansh Arpit, Jacek Tabor, Kyunghyun Cho*, and Krzysztof Geras*. The break-even point on optimization trajectories of deep neural networks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=r1g87C4KwB.

[9] Ryo Karakida and Shotaro Akaho. Learning curves for continual learning in neural networks: Self-knowledge transfer and forgetting, 2022. URL https://arxiv.org/abs/2112.01653.

[10] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International conference on machine learning*, pages 3519–3529. PMLR, 2019.

[11] Daniel S. Park, Jascha Sohl-Dickstein, Quoc V. Le, and Samuel L. Smith. The effect of network width on stochastic gradient descent and generalization: an empirical study, 2019. URL https://arxiv.org/abs/1905.03776.

[12] Haozhe Shan and Blake Bordelon. A theory of neural tangent kernel alignment and its influence on training. *arXiv preprint arXiv:2105.14301*, 2021.

[13] Greg Yang. Tensor programs ii: Neural tangent kernel for any architecture, 2020. URL https://arxiv.org/abs/2006.14548.

Appendix A. The NTK Framework and NTK Spectrum

Consider a dataset $\{\mathbf{x}_i\}_{i=1}^n$ with real targets $\{y_i\}_{i=1}^n$ and loss function ℓ. Let $f_t := f(\theta_t)$ be the neural network at time t, with parameters θ. By gradient descent, in continuous time, the parameters evolve as:

$$\partial_t\theta(t) = -\nabla_\theta \ell = -\frac{1}{n}\sum_{i=1}^{n} \frac{\partial \ell}{\partial f_t(\mathbf{x_i})} \cdot \frac{\partial f_t(\mathbf{x_i})}{\partial \theta_t} \tag{4}$$

Hence, by chain rule the network function space is determined by the *Neural Tangent Kernel* $\Theta_t(\mathbf{x}_i, \mathbf{x}_j) = \nabla_\theta f_t(\mathbf{x}_i)^\top \nabla_\theta f_t(\mathbf{x}_j)$:

$$\partial_t f_t(\mathbf{x}) = \frac{\partial f_t}{\partial \theta_t} \cdot \partial_t \theta_t = -\frac{1}{n}\sum_{i=1}^{n} \Theta_t(\mathbf{x}, \mathbf{x}_i) \frac{\partial \ell}{\partial f_t(\mathbf{x_i})} \tag{5}$$

A.1. Eigenvalues of NTK

Define the function output on dataset as $\mathbf{f} = [f(\mathbf{x_1}), f(\mathbf{x_2}), ..., f(\mathbf{x_n})]^\top$ and define the residuals as $\mathbf{e} = \mathbf{f} - \mathbf{y}$. For squared loss, the evolution of $\mathbf{f}$ and $\mathbf{e}$ at each time step is:

$$\mathbf{e}_{t+1} = (\mathbf{I} - \eta\Theta_t)\mathbf{e}_t, \tag{6}$$

We diagonalize the NTK as $\Theta = Q\Lambda Q^\top$ and project $\mathbf{e}$ onto the eigenbasis Q:

$$\tilde{\mathbf{e}}_{t+1} = (\mathbf{I} - \eta\Lambda)\tilde{\mathbf{e}}_t, \quad \tilde{e}^i_{t+1} = (1 - \eta\lambda_i)\tilde{e}^i_t \quad \text{for each eigenmode } i \tag{7}$$

Thus, the error in each eigenmode decays at a rate determined by the corresponding eigenvalue λ_i, indicating that the NTK governs the learning speed of each mode based on its eigenvalue. An higher NTK norm thus corresponds to faster convergence in some eigenmodes.

Appendix B. Experiment details

B.1. Task shifts experiments

To analyze learning dynamics in a continual learning setting, we train a simple Convolutional Neural Network (CNN) consisting of three convolutional layers, three pooling layers, and a fully connected layer with ReLU activation functions for image classification on CIFAR and ImageNet. All experiments use the SGD optimizer and cross-entropy loss. Neural Tangent Kernel (NTK) matrices are computed based on Yang [13], using a batch size of 32 random samples.

The classification problem on CIFAR-10 is split into two tasks, each containing five classes. We explore various settings, including CNN widths (64, 128, 256, 512, 1024, 2048), learning rates (1e-3, 1e-4, 1e-5), and training epochs (10, 20, 40, 80, 160). Here, the width of the CNN refers to the number of channels in the convolutional layers.

B.2. Task Similarity Experiments

Experiment 1: Gradual Shift with New Class Introduced. We define a family of input distributions $\mathcal{D}_i = \{i, i+1, \ldots, i+9\}$, where each $\mathcal{D}_i$ is a uniform mixture over 10 consecutive CIFAR-100 classes. In experiment E_i, we construct a two-task continual learning scenario: **Task 1** trains on $\mathcal{D}_0$ and **Task 2** trains on $\mathcal{D}_i$ for $i = 0, \ldots, 10$. The similarity between $\mathcal{D}_0$ and $\mathcal{D}_i$ is defined as:

$$\text{Similarity}(\mathcal{D}_0, \mathcal{D}_i) = \frac{|\mathcal{D}_0 \cap \mathcal{D}_i|}{|\mathcal{D}_0 \cup \mathcal{D}_i|}.$$

Experiment 2: Gradual Shift within Fixed Class Support. Define two disjoint sets of classes $\tilde{\mathcal{D}}_0 = \{0, 1, 2, 3, 4\}$ and $\tilde{\mathcal{D}}_1 = \{5, 6, 7, 8, 9\}$, and construct a family of mixed distributions:

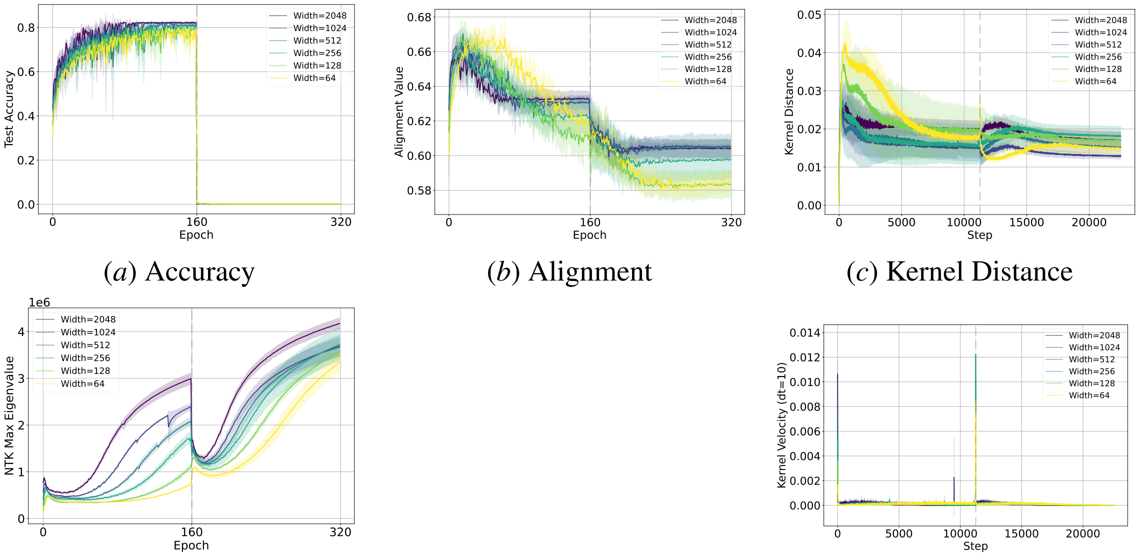
$$\tilde{\mathcal{D}}_\alpha = (1 - \alpha)\tilde{\mathcal{D}}_0 + \alpha\tilde{\mathcal{D}}_1, \quad \alpha \in \{0.1, 0.2, \ldots, 0.9\}.$$

Here for each experiment $\tilde{E}_\alpha$, **Task 1** is fixed to learn from $\tilde{\mathcal{D}}_{0.1}$, and **Task 2** learns from $\tilde{\mathcal{D}}_\alpha$.

Appendix C. Complementary Results

C.1. Comprehensive Metrics Visualization for CIFAR10 Two-Task Learning

In Figures 4, 5, and 6, we show various metrics calculated during the CIFAR10 experiment as described in B.1.



(*a*) Accuracy

(*b*) Alignment

(*c*) Kernel Distance

(*d*) Max Eigenvalue

(*e*) Velocity (dt=10)

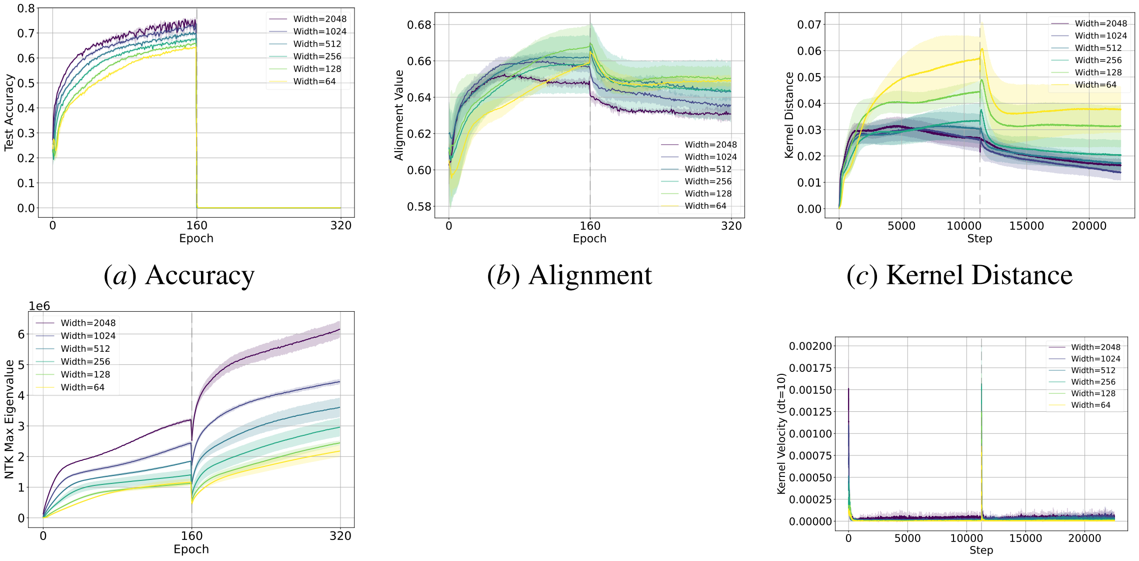
Figure 4: Comparison of different metrics across network widths for CNN trained on CIFAR10 with learning rate 0.001. The number of epochs per task is set to 160. (a) Test accuracy, (b) Alignment, (c) Kernel distance, (d) Maximum eigenvalue of NTK, (e) Kernel velocity with dt=10.

C.2. Metrics Visualization for Multiple Task Switches

To investigate whether the patterns persist during different task switches, we also perform experiments on 5 sequential tasks with 2 classes in each task on CIFAR-10 shown in Figure 7.

C.3. Experiments on ImageNet100

To further support our conclusions, we analyzed the evolution of the NTK spectrum on a larger dataset, ImageNet100. In Figure 8, we compare the effects of varying network width and the number of epochs per task on the NTK spectrum.



(*a*) Accuracy (*b*) Alignment (*c*) Kernel Distance

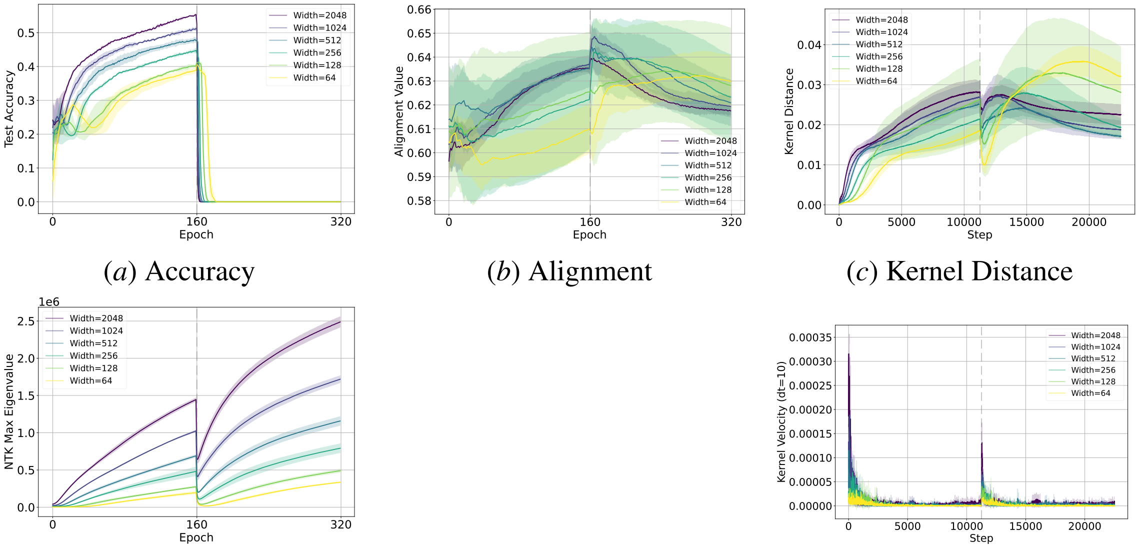
(*d*) Max Eigenvalue (*e*) Velocity (dt=10)

Figure 5: Comparison of different metrics across network widths for CNN trained on CIFAR10 with learning rate 0.0001. The number of epochs per task is set to 160. (a) Test accuracy, (b) Alignment, (c) Kernel distance, (d) Maximum eigenvalue of NTK, (e) Kernel velocity with dt=10.

All experiments use a learning rate of 1×10^{-3} and SGD as the optimizer. Comparisons are made relative to the base setting: width 250 and 10 epochs per task.

C.4. Standard Parametrization Results

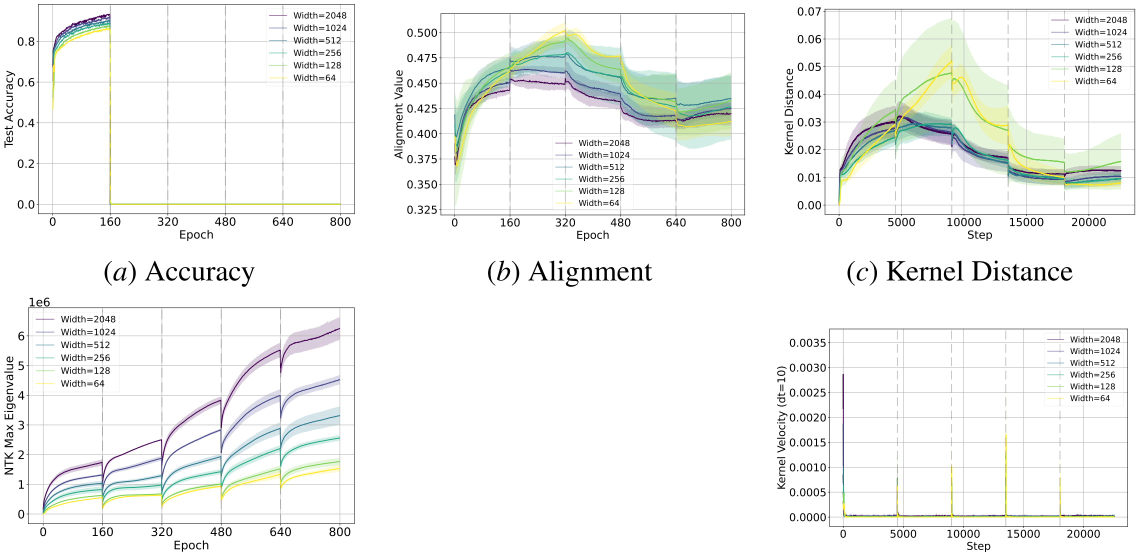
In order to guarantee stability as we scale the model width we adopt the standard parametrization using Kaiming Normal initialization and scale the learning rate with respect to width with $0.1/width$ to ensure stable training [11].The addition results presented in Figure 9 demonstrate that as the network width increases exponentially from 64 to 2048, the magnitude of changes in test accuracy (a), alignment (b), kernel distance (c), and the maximum eigenvalue of the NTK (d) decreases during task transitions in continual learning.



(*a*) Accuracy (*b*) Alignment (*c*) Kernel Distance

(*d*) Max Eigenvalue (*e*) Velocity (dt=10)

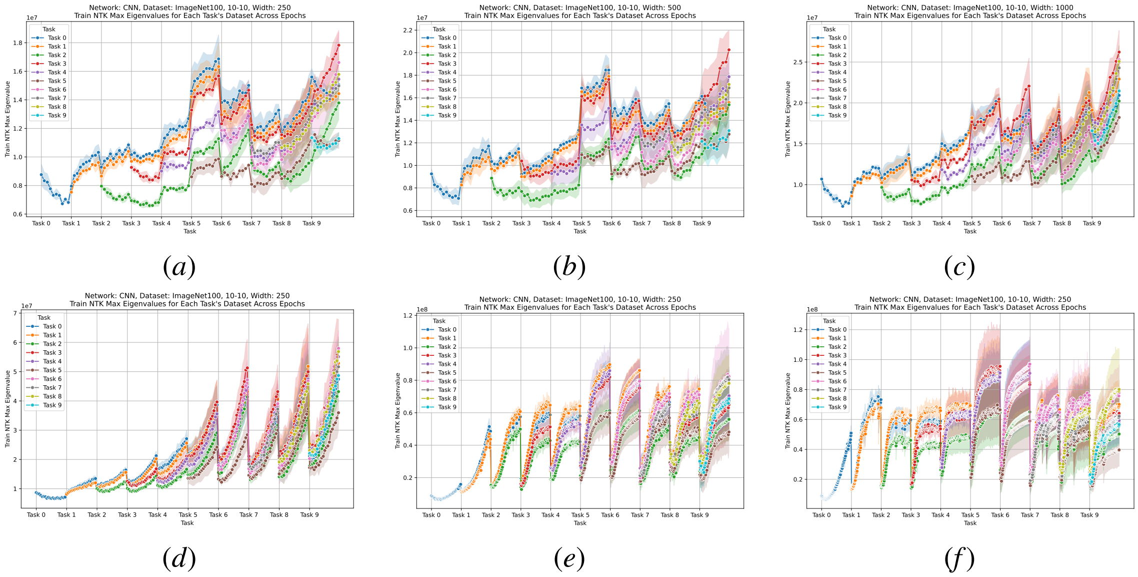
Figure 6: Comparison of different metrics across network widths for CNN trained on CIFAR10 with learning rate 0.00001. The number of epochs per task is set to 160. (a) Test accuracy, (b) Alignment, (c) Kernel distance, (d) Maximum eigenvalue of NTK, (e) Kernel velocity with dt=10.



(*a*) Accuracy (*b*) Alignment (*c*) Kernel Distance

(*d*) Max Eigenvalue (*e*) Velocity (dt=10)

Figure 7: Comparison of different metrics across network widths for CNN trained on CIFAR10 during multiple task switches. The number of epochs per task is set to 160. (a) Test accuracy, (b) Alignment, (c) Kernel distance, (d) Maximum eigenvalue of NTK, (e) Kernel velocity with dt=10.



(*a*)

(*b*)

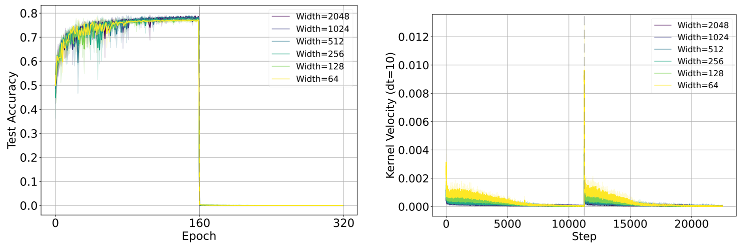
(*c*)

(*d*)

(*e*)

(*f*)

Figure 8: The effect of width and number of epochs per task on NTK spectrum on ImageNet100. The first row, left to right width 250, 500 and 1000 respectively, the second row, left to right epoch number per task 20, 50 and 100 respectively.



(*a*) Accuracy

(*b*) Velocity (dt=10)

Figure 9: Additional metrics across network widths for CNN trained on CIFAR10 The number of epochs per task is set to 160. (a) Test accuracy, (b)) Kernel velocity with dt=10.