
The Graphon Limit Hypothesis: Understanding Neural Network Pruning via Infinite Width Analysis

Anonymous Author(s)

Affiliation

Address

email

Abstract

1 Sparse neural networks promise efficiency, yet training them effectively remains a
2 fundamental challenge. Despite advances in pruning methods that create sparse
3 architectures, understanding why some sparse structures are better trainable than
4 others with the same level of sparsity remains poorly understood. Aiming to de-
5 velop a systematic approach to this fundamental problem, we propose a novel
6 theoretical framework based on the theory of graph limits, particularly graphons,
7 that characterizes sparse neural networks in the infinite-width regime. Our key
8 insight is that connectivity patterns of sparse neural networks induced by pruning
9 methods converge to specific graphons as networks' width tends to infinity, which
10 encodes implicit structural biases of different pruning methods. We postulate the
11 *Graphon Limit Hypothesis* and provide empirical evidence to support it. Leveraging
12 this graphon representation, we derive a *Graphon Neural Tangent Kernel (Graphon*
13 *NTK)* to study the training dynamics of sparse networks in the infinite width limit.
14 Graphon NTK provides a general framework for the theoretical analysis of sparse
15 networks. We empirically show that the spectral analysis of Graphon NTK corre-
16 lates with observed training dynamics of sparse networks, explaining the varying
17 convergence behaviours of different pruning methods. Our framework provides
18 theoretical insights into the impact of connectivity patterns on the trainability of
19 various sparse network architectures.

20 1 Introduction

21 Deep neural networks have achieved remarkable success in a wide range of machine learning tasks,
22 including computer vision [32, 17], natural language processing [61, 16], and scientific modeling [37].
23 A key ingredient in this success is overparameterisation [4, 9, 36, 5], networks are often trained with
24 many more parameters than are strictly necessary, which facilitates optimization and generalization.
25 However, this overparameterisation introduces substantial computational and memory overheads
26 [24, 59, 55, 67], hindering deployment in resource-constrained environments such as mobile/edge
27 devices, embedded systems, or real-time applications.

28 To address these challenges, network pruning has emerged as a fundamental approach for compressing
29 deep networks by removing redundant weights [40, 30, 22, 24]. By identifying and eliminating
30 parameters that contribute little to the model's output, pruning can produce sparse subnetworks
31 that retain high performance while offering significant efficiency gains. Empirically, the Lottery
32 Ticket Hypothesis (LTH) [22] demonstrates that randomly initialised dense networks contain sparse
33 subnetworks (winning tickets) that, when trained in isolation, can match or exceed the performance
34 of the original network. This observation has sparked significant research efforts including iterative
35 pruning methods for finding such winning tickets [23, 14, 13, 27], post-training pruning approaches
36 [28], various Pruning at Initialisation (PaI) techniques [42, 60, 64, 55, 67], and dynamic sparse

training [50, 20, 43, 44]. Parallel to this empirical progress, theoretical work has sought to prove the existence of effective sparse neural networks [48, 54, 11, 10]. Other works [47, 21] attempt to understand the effectiveness of sparse neural networks by analyzing gradient flow during training. Through the lens of the Ramanujan graph, [62, 7, 34] indicate that maximizing the graph connectivity of sparse networks improves the performance of subnetworks. Recently, [72] have studied the training dynamics of randomly pruned networks via NTK in an infinitely wide neural network setting.

Despite these advances, a comprehensive theoretical framework for understanding neural network pruning and the training dynamics of general sparse neural networks remains elusive. A particularly challenging aspect is explaining why sparse networks are often difficult to train effectively [25]. In this paper, we aim to develop a systematic framework for analysing sparse networks obtained by PaI methods. While NTK and other large width limit approaches [36, 5, 41] provide valuable tools for analysing dense neural network training, they are not suitable for the sparse network settings due to a fundamental difficulty of defining the limit of sparse networks of different sizes.

To address this problem, we develop a novel approach by leveraging graph limit theory. In particular, a pruning method produces binary masks that determine the active connections between neurons in layers. These masks can naturally be interpreted as the adjacency matrices of graphs defined over neural network layers. As the width of the network increases, the size of these adjacency matrices grows, and their structure becomes more regular and well-defined. In this view, the sequence of pruning masks generated by increasingly wide networks forms a sequence of graphs.

We hypothesize that, in the infinite-width limit, this sequence of graphs converges to a graphon that serves as a limit object for dense or structured sparse graphs [46, 8]. Informally, any large graph can be viewed as a random sample from some underlying graphon $\mathcal{W}(u, v)$ encoding the probability of an edge between node indices u and v [56]. We propose the **Graphon Limit Hypothesis** for sparse neural networks obtained by a given pruning method in the large width limit and provide empirical evidence to support it. Informally, the hypothesis states that:

Given a sparsity level, each network pruning method defines sequences of binary masks $(M_n^{(l)})$ that converges layer-wisely to graphons $\mathcal{W}^{(l)}$ in cut distance as the network width $n \rightarrow \infty$.

This new perspective allows us to unify diverse pruning techniques under a common mathematical framework. For example, Random Pruning at initialisation produces subnets whose configuration is an Erdős–Rényi random graph, hence its graphon limit corresponds to a constant graphon, while other PaI methods produce graphons with different patterns. Crucially, these graphons can be used to define an infinite-width model, analogous to how fully-connected networks yield NTK in the limit [36, 5, 41]. We formalise this idea by introducing the **Graphon Neural Tangent Kernel** (Graphon NTK), a kernel that captures the infinite-width behaviour of a pruned network specified by a graphon.

Our framework offers a new approach for analysing sparse neural networks through their graphon limits, provides a theoretically grounded tool for understanding how pruning affects training dynamics, and offers a principled basis for comparing or designing pruning strategies via their associated NTK behaviour. This framework generalises previous work on randomly pruned networks [72] where Random Pruning corresponds to constant graphons. Our contributions are summarised as follows:

- We introduce the concept of graphon limits of pruning masks and propose the *Graphon Limit Hypothesis* that each pruning method corresponds to a distinct graphon in the infinite-width limit, supported by empirical evidence showing that pruning masks exhibit structural convergence and generate characteristic graphons.
- We formalise the Graphon NTK, a new kernel framework that combines pruning structure with infinite-width analysis, and show how it can be derived from the graphon representation of the mask, highlighting the key differences between NTK and Graphon NTK.
- We empirically establish a connection between the spectral properties of the Graphon NTK and training dynamics of sparse networks, providing theoretical insights into how connectivity patterns affect the trainability of sparse networks.

The remainder of this paper is organised as follows: Sections 2, 3 discuss related works and provide preliminaries on graph limits, NTK, and pruning methods, respectively. In Section 4, we formalise the Graphon Limit Hypothesis and provide empirical evidence for its validity. Section 5 derives the Graphon NTK and its special case of Random Pruning. Section 6 presents our experimental results. Section 7 concludes with a summary of our contributions and directions for future research.

2 Related work

Neural network pruning. Pruning has been extensively studied as a means of reducing the computational and memory demands of deep networks. Early works propose magnitude-based pruning, where weights with the smallest absolute values are removed after training [40, 31, 30]. The Lottery Ticket Hypothesis (LTH) [22] represents a paradigm shift by demonstrating that dense networks contain sparse subnetworks that can be trained in isolation to match or exceed the performance of the original network. Follow-up works expand this observation across architectures [14, 13] and training regimes [23, 27], while theoretical analyses [48, 54, 52, 10, 11] seek to establish formal conditions under which such "winning tickets" exist. However, finding these winning tickets typically requires computationally intensive iterative train-prune-retrain cycles. To address this computational burden, Pruning at Initialisation (PaI) methods have emerged, which identify sparse subnetworks before training begins. Subnetworks can be found based on the magnitude, gradient, or hessian information [42, 63, 15, 3] or NTK-based scores [45, 53, 29, 65], while other methods [60, 55, 67] try to identify these tickets based on the subnetworks' configuration only. Despite these advances, analyses of why different pruning methods yield varying performance at the same sparsity level remain largely empirical. Works such as [47, 21, 35] have analysed gradient flow in sparse networks, while [72] examines the NTK limit behaviour of randomly pruned networks. Our work advances these efforts by providing a theoretical framework for understanding the limiting behaviour of pruned networks as their width increases.

Graphon and graph limit. Graphons are limit objects of graph sequences and have become a central tool in the study of large networks [46, 8]. A graphon is a symmetric measurable function $\mathcal{W} : [0, 1]^2 \rightarrow [0, 1]$ that can be viewed as the continuum analogue of an adjacency matrix. Graphons have been widely used in, e.g., modelling social networks [58, 57], and community detection [38, 1]. In machine learning, graphons have found use in graph neural network (GNN) analysis. Graphon neural networks extend traditional GNNs by considering their behaviour in the limit of large graphs [56, 51, 49, 33]. In particular, graphon neural networks operate on functions over $[0, 1]$ rather than on finite-dimensional vectors, which makes it possible to model infinitely large graphs or to generalize across graphs of different sizes. Another line of work focuses on learning a graphon from a collection of observed graphs [2, 12, 68, 66]. Specifically, to estimate graphons, SAS [12] reorders adjacency matrices by degree before applying smoothing, SBA [2] fits a stochastic block model to the graph data, GWB [68] relaxes the cut distance by Gromov-Wasserstein distance and minimizes this distance between observed graphs, and IGNN [66] directly uses neural networks. However, to the best of our knowledge, no prior work has connected graphons to the behaviour of pruning methods in neural networks. Our work introduces a novel interpretation: pruning masks of increasingly wide neural networks define a sequence of binary graphs that converge to graphons, and these graphons encode structural priors in the infinite-width limit.

Neural tangent kernel and infinite-width limit. The neural tangent kernel (NTK) [36, 5, 41, 18] characterises the training dynamics of infinitely wide neural networks under gradient descent. In this regime, training a network becomes equivalent to solving a kernel regression problem using the NTK. The NTK has since been extended and studied across a wide variety of architectures, including CNNs, RNNs, transformers, GNNs [5, 69, 70, 39, 19]. Among these, Graph Neural Tangent Kernels (GNTKs) [19] allow study on the limiting behaviour of infinitely wide (the number of features) GNNs trained via gradient descent. Later, [39] demonstrates that as the size of the graph increases, the GNTK converges to a graphon neural tangent kernel¹. While most NTK analyses assume dense architectures, recent studies have begun to explore sparse NTK models. [72] shows that Random Pruning preserves the NTK in the infinite-width limit up to a scaling factor, with convergence improving at larger widths. In particular, they apply pre-defined random masks on layers' weights to derive the NTK. Separately, [71] analyses sparsity induced by large bias initialisation, demonstrating that it leads to structured sparse activations and improved NTK conditioning, yielding faster convergence and tighter generalization bounds. Our work extends this line of research by developing a framework for analysing arbitrary pruning methods through their limiting graphons. This enables a more nuanced understanding of how different pruning strategies affect network trainability in the large width limit.

¹This kernel, also named Graphon NTK in [39], is for graph data and is different from our kernel.

3 Preliminaries

3.1 Neural tangent kernel

The neural tangent kernel (NTK) [36, 18, 5, 41] provides a theoretical framework for understanding the training dynamics of overparameterized neural networks. For a network $f(x; \theta)$ mapping input $x \in \mathbb{R}^d$ to output in \mathbb{R} , the NTK is defined as: $\Theta(x, x') = \nabla_{\theta} f(x, \theta)^{\top} \nabla_{\theta} f(x', \theta)$.

In the limit where the width of each hidden layer tends to infinity and the parameters are initialised randomly (e.g., with Gaussian scaling), the NTK converges to a deterministic kernel $\Theta_0(x, x')$, independent of training: for any given $t \geq 0$, $\Theta_t(x, x') \rightarrow \Theta_0(x, x')$, as $n \rightarrow \infty$. More recent works have shown that even networks with finite width, and any depth follows the NTK behaviour [5, 41]. Under gradient flow on the mean squared error loss, the predictions evolve according to a linear differential equation, which admits the solution: $f(X; \theta_t) = f(X; \theta_0)e^{-\eta\Theta_0 t} + (I - e^{-\eta\Theta_0 t}) \hat{y}$, where \hat{y} is the training label. This formula shows that the network predictions interpolate exponentially toward the training labels \hat{y} , with convergence behaviour governed by the spectral properties of Θ_0 . In particular, convergence is faster along directions corresponding to larger eigenvalues of the NTK.

Different architectures yield different limiting NTKs. For example, fully connected ReLU networks admit closed-form recursive formulas for the entries of Θ_0 [36], while convolutional and residual architectures induce structured NTKs that encode translation invariance and hierarchical composition [5]. Thus, the NTK framework bridges deep learning and kernel methods, offering theoretical insights into optimisation and generalisation.

3.2 Graphon and graph limit theory

Graph limit theory provides an analytic framework for analysing large graphs. For a sequence of graphs with an increasing number of nodes (G_n), one can often associate a limit object known as a graphon which is a symmetric, measurable function $\mathcal{W} : [0, 1]^2 \rightarrow [0, 1]$ that encodes the connectivity pattern between an infinite set of nodes [46, 8]. The function value $\mathcal{W}(x, y)$ represents the probability of an edge between nodes indexed by x and y in the limit. Graphon theory is particularly useful for capturing the limiting behaviour of a graph sequence (G_n) as the number of nodes $n \rightarrow \infty$.

Each finite graph G_n with n vertices can be represented (up to relabelling) by a step function \mathcal{W}_{G_n} on $[0, 1]^2$, where the unit interval is partitioned into n equal parts and edge presence is encoded by: $\mathcal{W}_{G_n}(x, y) = A_{ij}, \forall (x, y) \in [\frac{i-1}{n}, \frac{i}{n}) \times [\frac{j-1}{n}, \frac{j}{n})$, with A being the adjacency matrix of G_n . A sequence (G_n) is said to converge to a graphon \mathcal{W} if the sequence (\mathcal{W}_{G_n}) converges to \mathcal{W} in the labelled cut distance:

$$d_{\square}(\mathcal{U}, \mathcal{W}) = \sup_{S, T \subseteq [0, 1]} \left| \int_{S \times T} (\mathcal{U}(x, y) - \mathcal{W}(x, y)) dx dy \right|,$$

where \mathcal{U} and \mathcal{W} are two graphons, and the supremum is taken over all measurable subsets $S, T \subseteq [0, 1]$. To account for vertex relabelling, we define the cut distance between two graphons up to measure-preserving transformations. Let Φ be the set of all measure-preserving bijections $\phi : [0, 1] \rightarrow [0, 1]$. The (label-invariant) cut distance between graphons \mathcal{U} and \mathcal{W} is modified to be: $\delta_{\square}(\mathcal{U}, \mathcal{W}) = \inf_{\phi \in \Phi} d_{\square}(\mathcal{U}, \mathcal{W}^{\phi})$ where $\mathcal{W}^{\phi}(x, y) = \mathcal{W}(\phi(x), \phi(y))$. A sequence of graphs (G_n) with $|V(G_n)| \rightarrow \infty$ is said to converge to a graphon \mathcal{W} if $\lim_{n \rightarrow \infty} \delta_{\square}(\mathcal{W}_{G_n}, \mathcal{W}) = 0$.

The cut distance quantifies the similarity between two graph adjacency matrices or functions by considering all possible labellings of their nodes [46]. Intuitively, two graphs are close in the cut distance if their global connectivity structures are similar. An equivalent characterization of the convergence of a graph sequence (G_n) to a graphon is based on homomorphism counting [8, 56]. Particularly, for every finite subgraph F , the homomorphism densities $t(F, G_n)$ converge to $t(F, \mathcal{W})$. The homomorphism density $t(F, G)$ represents the probability that a random map from the vertices of F to those of G preserves all edges.

4 Graph limit and sparse neural networks

Neural network pruning produces binary masks over the network's layers to specify which weights are removed from each layer. These masks naturally define bipartite graphs between adjacent layers, with the mask matrix $M^{(l)} \in \{0, 1\}^{n_{l-1} \times n_l}$ acting as the biadjacency matrix. We hypothesize that

pruning-induced structures can be modelled in the infinite-width limit using graphons, enabling a unified geometric and functional interpretation of sparse neural networks.

4.1 Pruning masks as graphs

In a fully connected network, each neuron in layer $l - 1$ is connected to every neuron in layer l , resulting in a dense weight matrix $W^{(l)} \in \mathbb{R}^{n_{l-1} \times n_l}$. A pruning method replaces this weight matrix with a masked version $\tilde{W}^{(l)} = W^{(l)} \odot M^{(l)}$ where $M^{(l)} \in \{0, 1\}^{n_{l-1} \times n_l}$ is the binary mask and \odot denotes elementwise multiplication. The binary mask $M^{(l)}$ defines a bipartite graph, and in the infinite-width limit, we model it with a bipartite graphon $\mathcal{W}^{(l)} : [0, 1]^2 \rightarrow [0, 1]$. From this point onward, we use \mathcal{W} to denote a bipartite graphon, unless explicitly mentioned otherwise.

Despite being called “sparse”, most pruning methods retain a constant fraction of weights (e.g., 10%), resulting in $\Theta(n^2)$ connections for width n (although with very small constants). From a graph-theoretic perspective, these remain dense graphs, where graphon theory applies. Thus, to ensure well-defined graphon limits, we assume: (i) the pruning masks retain a constant fraction $p > 0$ of weights (normally satisfied in pruning context), and (ii) the width of hidden layers remains comparable across layers as $n \rightarrow \infty$. Under these assumptions, given a pruning method, for a sequence of pruning masks $(M_n^{(l)})$ with increasing width, we can define their convergence to a bipartite graphon $\mathcal{W}^{(l)}$ analogously: $\lim_{n \rightarrow \infty} \delta_{\square}(\mathcal{W}_{M_n^{(l)}}, \mathcal{W}^{(l)}) = 0$ where $\mathcal{W}_{M_n^{(l)}}$ is the bipartite graphon representation of the mask $M_n^{(l)}$ at layer l (see Section 3.2 for definition of graphon representation of graphs). As the network width $n \rightarrow \infty$, each mask matrix becomes a larger and more structured graph. Different pruning methods induce different sequences $(M_n^{(l)})$ of such biadjacency matrices.

4.2 Graphon Limit Hypothesis for neural network pruning

Graphon Limit Hypothesis. *Given a sequence of neural networks (N_n) from a fixed architecture class \mathcal{A} with widths tending to infinity, the application of a pruning method \mathcal{P} at fixed sparsity level $p > 0$ produces sequences of binary masks $(M_n^{(l)})$ that converge layer-wisely to deterministic graphons $\mathcal{W}^{(l)}$ in the cut distance. These limiting graphons depend only on $\mathcal{A}, p, \mathcal{P}$ and characterize the connectivity structure of the pruned networks in the infinite-width limit.*

The graphon limit has the following geometry interpretation. A fundamental result from graph limit theory states that a sequence of graphs (G_n) converges to a graphon \mathcal{W} if and only if the density of any fixed subgraph F in (G_n) converges to the density of F in \mathcal{W} . In the context of neural networks, this means the graphon limit asymptotically captures geometric patterns including path densities (effective paths), and other structural motifs that influence the network’s computational properties.

This hypothesis has several important implications: (i) graphons $\mathcal{W}^{(l)}$ serve as structural priors for the pruned network in the infinite-width setting; (ii) each pruning method corresponds to a distinct region in the space of graphons, determined by the method’s design; and (iii) structural differences between pruning methods can be captured and compared through their limiting graphons.

4.3 Experiments on graph limit of pruning at initialisation methods

We empirically validate the Graphon Limit Hypothesis by examining whether pruning methods converge to distinct, characteristic graphons as network width increases. We analyse four pruning at initialisation methods: Random, SNIP [42], GraSP [63], and Synflow [60] across varying network widths $n \in \{100, 500, 1000, 2000\}$, number of layers $\{4, 5\}$, and sparsity levels $\{70\%, 80\%, 90\%\}$. We conduct 100 independent trials per configuration and collect layer masks except for masks from input and output layers. To visualise the emergent graphons, we employ the SAS method [12] that: (1) Sorts nodes based on degree centrality (out-degree for layer l , in-degree for layer $l + 1$); (2) Partitions the sorted bipartite graph into a grid of intervals; (3) Computes the average edge density within each interval. This degree-based sorting serves as an approximate measure-preserving transformation, revealing underlying structural patterns while maintaining invariance to node permutations. We refer to Appendix C for more details.

Figure 1 displays the estimated graphons with increasing network widths. Each pruning method converges to a distinct pattern. In particular, Random Pruning converges to a constant graphon

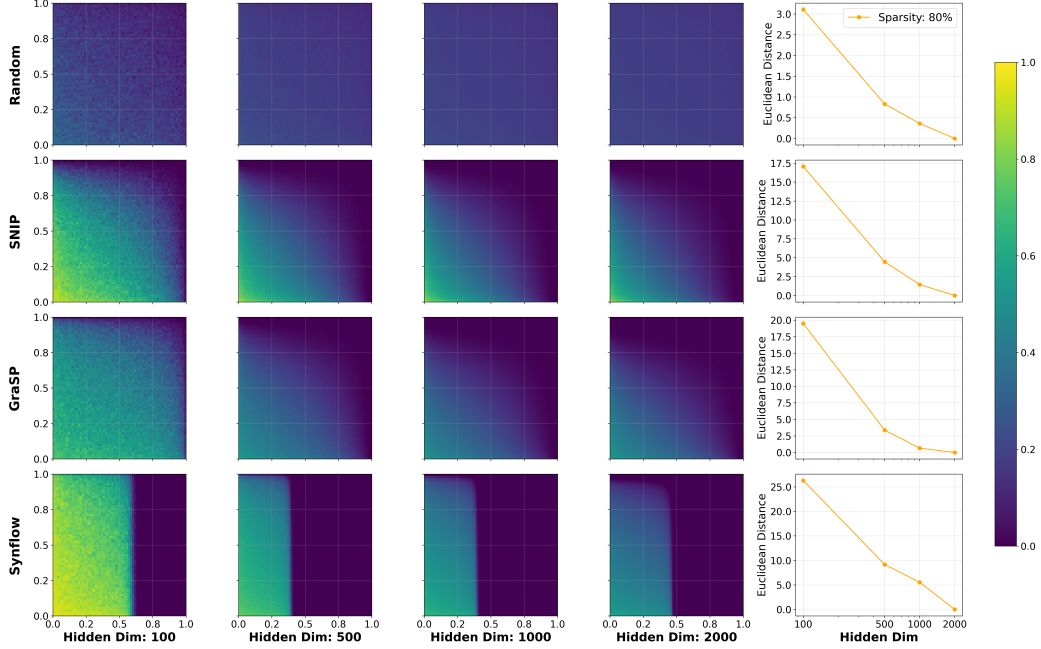


Figure 1: Graph limit of subnetworks’ mask produced by PaI methods at 80% sparsity and the corresponding convergence of graphons via Euclidean distances.

(Erdős-Rényi random graph), with uniform connection probability across all node positions. SNIP and GraSP exhibit structured, non-uniform graphons with density gradients, preferentially connecting high-centrality nodes. Synflow converges to a block-like graphon with sharp transitions, strongly prioritising connections among high-centrality neurons. This method encourages sparse networks with a high number of paths, which is also indicated in [55, 53].

To quantify convergence, we also show the Euclidean distance between density matrices at width n and reference matrices at $n = 2000$ in Figure 1-right. Since all histograms are aligned via degree-based sorting, we use the Euclidean distance between the density matrices as a proxy for the cut distance. All methods demonstrate monotonic convergence, with distances decreasing as width increases, confirming that the limiting graphon structure is an intrinsic characteristic of each pruning algorithm.

These results provide compelling evidence that each pruning method induces a unique (up to relabelling), stable connectivity pattern in the large-width limit, validating our graphon hypothesis and establishing a foundation for analysing pruning methods through graph limit theory.

5 Neural tangent kernel with graphon structure

In this section, we present the derivation of NTK for neural networks with graphon structure. This novel formulation extends the standard NTK theory to accommodate networks where connectivity patterns are modulated by graphon functions, providing insights into how sparse network architecture influences learning dynamics.

5.1 Network structure and setup

In standard neural networks, weights between layers are typically sampled from identical independent distributions. However, sparse networks exhibit connectivity patterns, where connection strength depends on neuron positions. Graphon provides a mathematical tool to model such connectivity patterns. Intuitively, we can view each neuron as having a position in $[0, 1]$, and the graphon value $\mathcal{W}^{(l)}(u, v)$ represents the expected connection strength/probability between neurons at positions u and v . Each layer’s structure is thus defined by its corresponding graphon $\mathcal{W}^{(l)}$, providing a continuous representation of the network’s connectivity.

Graphon-structured networks naturally connect to neural network pruning, where certain connections are eliminated. In pruning, a binary mask $M^{(l)}$ is applied to weights: $\widetilde{W}^{(l)} = W^{(l)} \odot M^{(l)}$. For weights initialised as $W_{ij}^{(l)} \sim \mathcal{N}(0, \sigma_w^2)$, pruning effectively modifies their variance: $\text{Var}(\widetilde{W}_{ij}^{(l)}) = \text{Var}(W_{ij}^{(l)} M_{ij}^{(l)}) = \text{Var}(W_{ij}^{(l)}) M_{ij}^{(l)} = \sigma_w^2 M_{ij}^{(l)}$.

Based on the Graphon Limit Hypothesis in Section 4, as network width increases, pruning masks converge to graphons layer-wisely $\mathcal{W}^{(l)}(u_l, u_{l-1})$, representing the probability density of connections between positions. In a finite network, the weights are sampled as: $W_{ij}^{(l)} \sim \mathcal{N}\left(0, \mathcal{W}^{(l)}\left(\frac{i}{n_l}, \frac{j}{n_{l-1}}\right)\right)$, where n_l represents the width of layer l , and we consider $\sigma_w^2 = 1$ for simplicity [5]. Similar to the standard NTK setting [36, 5], we remove the bias terms and consider a single output network. Unlike standard networks where weights are i.i.d., our graphon-modulated weights have position-dependent variances while maintaining independence. As we take layer widths to infinity $n_l \rightarrow \infty$, discrete neuron indices approach to continuous positions in $[0, 1]$: $i/n_l \rightarrow u_l \in [0, 1]$ (position in layer l) and $j/n_{l-1} \rightarrow u_{l-1} \in [0, 1]$ (position in layer $l-1$). Summations over neurons become integrals over positions, and the scaling term $1/n_{l-1}$ will be absorbed into the integral. The forward pass of an L hidden layers network below illustrates this approach from discrete to continuous:

Discrete network

$$z_i^{(l)}(x) = \frac{1}{\sqrt{n_{l-1}}} \sum_{j=1}^{n_{l-1}} W_{ij}^{(l)} h_j^{(l-1)}(x),$$

$$h_i^{(l)}(x) = \sigma(z_i^{(l)}(x)),$$

$$f(x) = \frac{1}{\sqrt{n_L}} \sum_{j=1}^{n_L} W_{ij}^{(L+1)} h_j^{(L)}(x),$$

Continuous network

$$z^{(l)}(u_l, x) = \int_0^1 W^{(l)}(u_l, u_{l-1}) h^{(l-1)}(u_{l-1}, x) du_{l-1},$$

$$h^{(l)}(u_l, x) = \sigma(z^{(l)}(u_l, x)),$$

$$f(x) = \int_0^1 W^{(L+1)}(u_{L+1}, u_L) h^{(L)}(u_L, x) du_L,$$

where $W^{(l)}(u_l, u_{l-1}) \sim \mathcal{N}(0, \mathcal{W}^{(l)}(u_l, u_{l-1}))$ is a random field modulated by the graphon. In addition, since the statistical properties of the network now is affected by the graphon structure, weights are independent but *not identically* distributed random variables. We need additional assumptions to ensure the Law of Large Numbers (LLN) and Central Limit Theorem (CLT) still apply. For the LLN to hold in this non-iid setting, we assume: (i) the graphon values $\mathcal{W}^{(l)}(u_l, u_{l-1})$ are bounded, and (ii) the average connectivity $\int_0^1 \mathcal{W}^{(l)}(u_l, u_{l-1}) du_{l-1}$ are well-behaved for all positions u_l . Similarly, for the CLT to apply to pre-activations, we assume the Lindeberg-Feller [6] condition:

$$\lim_{n_{l-1} \rightarrow \infty} \frac{1}{n_{l-1}} \sum_{j=1}^{n_{l-1}} \mathbb{E} \left[\left(W_{ij}^{(l)} h_j^{(l-1)}(x) \right)^2 \cdot \mathbf{1}_{\{|W_{ij}^{(l)} h_j^{(l-1)}(x)| > \epsilon \sqrt{n_{l-1}}\}} \right] = 0,$$

for all $\epsilon > 0$. This condition ensures that no single weight-activation product dominates the sum, allowing the pre-activations to still converge to Gaussian processes, albeit with position-dependent covariance structures shaped by the graphon.

5.2 Graphon neural tangent kernel

We extend the NTK theory to networks with graphon-structured connectivity. We begin by characterising the limiting behaviour of pre-activations and activations in the infinite-width limit.

Proposition 1. *For a neural network with layers structured by graphons $\mathcal{W}^{(l)} : [0, 1]^2 \rightarrow [0, 1]$, Lipschitz nonlinearity σ , and in the limit as $n_1, \dots, n_L \rightarrow \infty$, the pre-activations $z^{(l)}(u_l, x)$ at every hidden layer converge to centred Gaussian processes with covariance $\tilde{\Sigma}^{(l)}$, where $\tilde{\Sigma}^{(l)}$ is defined recursively by:*

$$\tilde{\Sigma}^{(1)}(u_1, u'_1, x, x') = \delta(u_1 - u'_1) \frac{1}{d} \sum_j^d \mathcal{W}^{(1)}(u_1, \frac{j}{d}) (x \cdot x')_j, \quad (1)$$

$$\tilde{\Sigma}^{(l)}(u_l, u'_l, x, x') = \delta(u_l - u'_l) \int_0^1 \mathcal{W}^{(l)}(u_l, u_{l-1}) \Sigma^{(l-1)}(u_{l-1}, u_{l-1}, x, x') du_{l-1}, \quad (2)$$

where $(x \cdot x')_j$ represents the input correlation at position j , the activation covariance $\Sigma^{(l)}$ is:

$$\Sigma^{(l)}(u_l, u'_l, x, x') = \delta(u_l - u'_l) \mathbb{E}_{(z, z') \sim \mathcal{N}(0, \Lambda^{(l)}(u_l))} [\sigma(z) \sigma(z')], \quad (3)$$

$\delta(u_l - u'_l)$ is the Dirac delta function, and $\Lambda^{(l)}(u_l)$ is the position-dependent covariance matrix:

$$\Lambda^{(l)}(u_l) = \begin{bmatrix} \tilde{\Sigma}^{(l)}(u_l, u_l, x, x) & \tilde{\Sigma}^{(l)}(u_l, u_l, x, x') \\ \tilde{\Sigma}^{(l)}(u_l, u_l, x', x) & \tilde{\Sigma}^{(l)}(u_l, u_l, x', x') \end{bmatrix}.$$

Remark 1. The key difference between the standard NTK and Graphon NTK lies in the pre-activation covariance formulation. In standard NTK, the pre-activation covariance $\tilde{\Sigma}^{(l)}$ equals the previous layer’s activation covariance $\Sigma^{(l-1)}$. In contrast, the Graphon NTK modulates this with the graphon function $\mathcal{W}^{(l)}$, creating position-dependent covariance structures. This causes signals to propagate non-uniformly through the network, with connectivity strength determined by the graphon values.

The NTK characterises how network outputs change with respect to parameters during training. Our key result shows that the Graphon NTK also converges to a deterministic limit in the infinite-width regime, but with a structure determined by the graphon connectivity patterns.

Theorem 1 (Graphon NTK). For a neural network with layers structured by graphons $\mathcal{W}^{(l)} : [0, 1]^2 \rightarrow [0, 1]$, Lipschitz nonlinearity σ , in the limit as $n_1, \dots, n_L \rightarrow \infty$, the Graphon Neural Tangent Kernel (Graphon NTK) $\Theta(x, x')$ converges to a deterministic kernel:

$$\Theta(x, x') = \sum_{l=1}^L \int_0^1 \left(\dot{\Sigma}^{(l)}(u_l, u_l, x, x') \int_{[0, 1]^{L-l+1}} \prod_{m=l+1}^{L+1} \mathcal{W}^{(m)}(u_m, u_{m-1}) \dot{\Sigma}^{(m)}(u_m, u_m, x, x') d\mathbf{u}_{l+1} \right) \cdot \left(\int_0^1 \Sigma^{(l-1)}(u_{l-1}, u_{l-1}, x, x') du_{l-1} \right) du_l, \quad (4)$$

where $\dot{\Sigma}^{(l)}(u_l, u_l, x, x') = \mathbb{E}[\sigma'(z^{(l)}(u_l, x)) \sigma'(z^{(l)}(u_l, x'))]$ represents the expected correlation between activation derivatives, and $d\mathbf{u}_{l+1} = du_{L+1} du_L \dots du_{l+1}$.

Remark 2. The Graphon NTK explicitly shows how graphon functions $\mathcal{W}^{(l)}$ shape the kernel through position-dependent connectivity, in contrast to the standard NTK. This structure gives insights on the heterogeneous learning dynamics of sparse model training, and allows modelling diverse connectivity patterns.

The Graphon NTK provides a powerful framework for analysing how the pattern of connectivity between neurons in sparse neural networks affects learning dynamics in the infinite-width limit. By modulating the weight variances according to the graphon, we can model a wide range of structured connectivity patterns, including those that arise from various network pruning strategies. We refer to Appendix A for detailed proofs.

5.3 Graphon neural tangent kernel of Random Pruning

Our Graphon NTK framework encompasses a broad class of structured sparsity patterns with Random Pruning being a special case. Specifically, when the underlying graphon is constant, i.e., $\mathcal{W}(u, v) = c$ where $c \in (0, 1]$, the resulting kernel simplifies to a scaled version of the standard NTK: $\Theta(x, x') = c^L \Theta_{\text{std}}(x, x')$, where Θ_{std} denotes the standard NTK of a fully-connected network with L hidden layers. This special case was previously studied in [72] by applying random masks to weight matrices. Our general framework of Graphon NTK not only recovers this result, but is also flexible enough to analyse more complex sparsity connectivity patterns beyond Random Pruning.

This scaling directly influences network training dynamics. If λ_k is the k -th eigenvalue of Θ_{std} , then this k -th eigenvalue of the pruned network’s NTK becomes $c^L \lambda_k$. Notably, while the absolute learning speed is reduced, the relative dynamics between modes remain unchanged. This offers a principled explanation for the empirical observation that sparse random networks converge more slowly than their dense counterparts [26]. We refer to Appendix B for more discussions.

6 Numerical experiments

We illustrate the relationship between spectral properties of Graphon NTK and training dynamics of finite sparse networks using three pruning methods: Random, SNIP [42], and Synflow [60] at sparsity

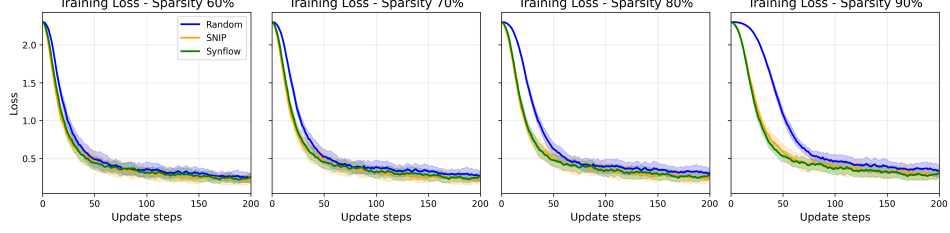


Figure 2: The training loss in the first 200 gradient update steps of training sparse networks produced by Random, SNIP, and Synflow with different sparsity levels.

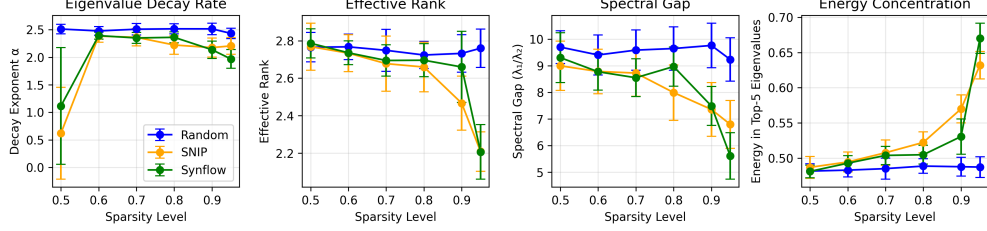


Figure 3: Spectral metrics of the Graphon NTK with different graphon functions and sparsity levels.

levels from 50% to 95%. Subnetworks are pruned from 4-layer networks with hidden size $n = 1024$, then trained on MNIST. We approximate the Graphon NTK by graphon functions found in Section 4. In particular, given the sparsity, we generate masks for 4-layer networks with hidden dimension $n = 1024$ based on graphon functions. Then we compute the Graphon NTK based on a batch of 128 data samples from 10 classes in MNIST, and analyse four spectral metrics: eigenvalue decay rate (α), effective rank $\text{trace}(\Theta)/\lambda_1$, spectral gap (λ_1/λ_2), and the energy concentration in top-5 eigenvalues $\frac{\sum_{i=1}^5 \lambda_i}{\sum_{j=1}^n \lambda_j}$. We refer to Appendix D for further details.

Results and discussion. Figure 3 reveals that Random Pruning maintains relatively consistent spectral properties across the sparsity levels, with stable decay rate, high effective rank, and broad spectral spread. This reflects uniform eigenvalue scaling which is consistent with the constant graphon analysis in Section 5.3, where Random Pruning acts as a global downscaling of the kernel. In contrast, SNIP and Synflow increasingly concentrate their Graphon NTK energy in top eigenvalues as the sparsity level grows, despite reduced effective rank and spectral gaps. This suggests a stronger focus on dominant eigen-directions, aligning with their faster training loss reduction at the beginning when compared with random subnetworks at the same sparsity in Figure 2.

The observed correlation between kernel spectral properties and training dynamics demonstrates an initial insight of our Graphon NTK framework for sparse neural network training. These findings offer both theoretical insight and practical guidance: Graphon NTK can serve as principled, training-free indicators for evaluating pruning quality, and preserving key spectral characteristics should be a design goal for future pruning algorithms, as also verified in [65].

7 Conclusion

In this paper, we introduce a novel theoretical framework for analysing sparse neural networks through the lens of graph limit theory and neural tangent kernel. Our Graphon Limit Hypothesis establishes a connection between pruning methods and their limiting graphons in the infinite-width regime, providing a mathematical framework for understanding the structural properties of sparse networks. We derive Graphon NTK which offers a meaningful tool for analyzing how these structural properties affect training dynamics of sparse networks. This paves the way for theoretical study of sparse models using tools from kernel and graph limit theories. Moreover, the framework can be used to guide the design of (i) new pruning algorithms by selecting or learning graphons with desirable kernel properties; and (ii) sparse training algorithms. It also leads to the possibility of performing sparsification directly by optimising over the space of graphons rather than discrete masks.

References

- [1] Emmanuel Abbe. Community detection and stochastic block models: recent developments. *Journal of Machine Learning Research*, 18(177):1–86, 2018.
- [2] Edo M Airolidi, Thiago B Costa, and Stanley H Chan. Stochastic blockmodel approximation of a graphon: Theory and consistent estimation. *Advances in Neural Information Processing Systems*, 26, 2013.
- [3] Milad Alizadeh, Shyam A. Tailor, Luisa M Zintgraf, Joost van Amersfoort, Sebastian Farquhar, Nicholas Donald Lane, and Yarin Gal. Prospect pruning: Finding trainable weights at initialization using meta-gradients. In *International Conference on Learning Representations*, 2022.
- [4] Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. *Advances in neural information processing systems*, 32, 2019.
- [5] Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Russ R Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. *Advances in neural information processing systems*, 32, 2019.
- [6] Krishna B Athreya and Soumendra N Lahiri. *Measure theory and probability theory*, volume 19. Springer, 2006.
- [7] Pal Bithika, Biswas Arindam, Kolay Sudeshna, Mitra Pabitra, and Basu Biswajit. A study on the ramanujan graph property of winning lottery tickets. In *International Conference on Machine Learning*, volume 162, pages 17186–17201, 2022.
- [8] Christian Borgs, Jennifer T Chayes, László Lovász, Vera T Sós, and Katalin Vesztegombi. Convergent sequences of dense graphs i: Subgraph frequencies, metric properties and testing. *Advances in Mathematics*, 219(6):1801–1851, 2008.
- [9] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- [10] Rebekka Burkholz. Convolutional and residual networks provably contain lottery tickets. In *International Conference on Machine Learning*, pages 2414–2433. PMLR, 2022.
- [11] Rebekka Burkholz. Most activation functions can win the lottery without excessive depth. *Advances in Neural Information Processing Systems*, 35:18707–18720, 2022.
- [12] Stanley Chan and Edoardo Airolidi. A consistent histogram estimator for exchangeable graph models. In *International Conference on Machine Learning*, pages 208–216. PMLR, 2014.
- [13] Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Michael Carbin, and Zhangyang Wang. The lottery tickets hypothesis for supervised and self-supervised pre-training in computer vision models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16306–16316, 2021.
- [14] Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. The lottery ticket hypothesis for pre-trained bert networks. *Advances in neural information processing systems*, 33:15834–15846, 2020.
- [15] Pau de Jorge, Amartya Sanyal, Harkirat Behl, Philip Torr, Grégory Rogez, and Puneet K. Dokania. Progressive skeletonization: Trimming more fat from a network at initialization. In *International Conference on Learning Representations*, 2021.
- [16] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186, 2019.

- [17] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.
- [18] Simon Du and Wei Hu. Width provably matters in optimization for deep linear neural networks. In *International Conference on Machine Learning*, pages 1655–1664. PMLR, 2019.
- [19] Simon S Du, Kangcheng Hou, Russ R Salakhutdinov, Barnabas Poczos, Ruosong Wang, and Keyulu Xu. Graph neural tangent kernel: Fusing graph neural networks with graph kernels. *Advances in neural information processing systems*, 32, 2019.
- [20] Utku Evci, Trevor Gale, Jacob Menick, Pablo Samuel Castro, and Erich Elsen. Rigging the lottery: Making all tickets winners. In *International Conference on Machine Learning*, pages 2943–2952. PMLR, 2020.
- [21] Utku Evci, Yani Ioannou, Cem Keskin, and Yann Dauphin. Gradient flow in sparse neural networks and how lottery tickets win. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 6577–6586, 2022.
- [22] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.
- [23] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel Roy, and Michael Carbin. Linear mode connectivity and the lottery ticket hypothesis. In *International Conference on Machine Learning*, pages 3259–3269. PMLR, 2020.
- [24] Elias Frantar and Dan Alistarh. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, pages 10323–10337. PMLR, 2023.
- [25] Advait Gadhikar, Tom Jacobs, Chao Zhou, and Rebekka Burkholz. Sign-in to the lottery: Reparameterizing sparse training from scratch. *arXiv preprint arXiv:2504.12801*, 2025.
- [26] Advait Gadhikar, Sree Harsha Nelaturu, and Rebekka Burkholz. Cyclic sparse training: Is it enough? *arXiv preprint arXiv:2406.02773*, 2024.
- [27] Advait Harshal Gadhikar and Rebekka Burkholz. Masks, signs, and learning rate rewinding. In *The Twelfth International Conference on Learning Representations*, 2024.
- [28] Trevor Gale, Erich Elsen, and Sara Hooker. The state of sparsity in deep neural networks. *arXiv preprint arXiv:1902.09574*, 2019.
- [29] Thomas Gebhart, Udit Saxena, and Paul Schrater. A unified paths perspective for pruning at initialization. *arXiv preprint arXiv:2101.10552*, 2021.
- [30] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28, 2015.
- [31] Babak Hassibi, David G Stork, and Gregory J Wolff. Optimal brain surgeon and general network pruning. In *IEEE international conference on neural networks*, pages 293–299. IEEE, 1993.
- [32] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [33] Daniel Herbst and Stefanie Jegelka. Higher-order graphon neural networks: Approximation and cut distance. In *The Thirteenth International Conference on Learning Representations*, 2025.
- [34] Duc N.M Hoang, Shiwei Liu, Radu Marculescu, and Zhangyang Wang. REVISITING PRUNING AT INITIALIZATION THROUGH THE LENS OF RAMANUJAN GRAPH. In *The Eleventh International Conference on Learning Representations*, 2023.
- [35] Tom Jacobs and Rebekka Burkholz. Mask in the mirror: Implicit sparsification. In *The Thirteenth International Conference on Learning Representations*, 2025.

- [36] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.
- [37] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *nature*, 596(7873):583–589, 2021.
- [38] Florian Klimm, Nick S Jones, and Michael T Schaub. Modularity maximization for graphons. *SIAM Journal on Applied Mathematics*, 82(6):1930–1952, 2022.
- [39] Sanjukta Krishnagopal and Luana Ruiz. Graph neural tangent kernel: Convergence on large graphs. In *International Conference on Machine Learning*, pages 17827–17841. PMLR, 2023.
- [40] Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. *Advances in neural information processing systems*, 2, 1989.
- [41] Jaehoon Lee, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. *Advances in neural information processing systems*, 32, 2019.
- [42] Namhoon Lee, Thalaiyasingam Ajanthan, and Philip Torr. Snip: Single-shot network pruning based on connection sensitivity. In *International Conference on Learning Representations*, 2019.
- [43] Shiwei Liu, Tianlong Chen, Xiaohan Chen, Zahra Atashgahi, Lu Yin, Huanyu Kou, Li Shen, Mykola Pechenizkiy, Zhangyang Wang, and Decebal Constantin Mocanu. Sparse training via boosting pruning plasticity with neuroregeneration. *Advances in Neural Information Processing Systems*, 34:9908–9922, 2021.
- [44] Shiwei Liu, Lu Yin, Decebal Constantin Mocanu, and Mykola Pechenizkiy. Do we actually need dense over-parameterization? in-time over-parameterization in sparse training. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 6989–7000. PMLR, 18–24 Jul 2021.
- [45] Tianlin Liu and Friedemann Zenke. Finding trainable sparse networks through neural tangent transfer. In *International Conference on Machine Learning*, pages 6336–6347. PMLR, 2020.
- [46] László Lovász and Balázs Szegedy. Limits of dense graph sequences. *Journal of Combinatorial Theory, Series B*, 96(6):933–957, 2006.
- [47] Ekdeep Singh Lubana and Robert Dick. A gradient flow framework for analyzing network pruning. In *International Conference on Learning Representations*, 2021.
- [48] Eran Malach, Gilad Yehudai, Shai Shalev-Schwartz, and Ohad Shamir. Proving the lottery ticket hypothesis: Pruning is all you need. In *International Conference on Machine Learning*, pages 6682–6691. PMLR, 2020.
- [49] Sohir Maskey, Ron Levie, and Gitta Kutyniok. Transferability of graph neural networks: an extended graphon approach. *Applied and Computational Harmonic Analysis*, 63:48–83, 2023.
- [50] Decebal Constantin Mocanu, Elena Mocanu, Peter Stone, Phuong H Nguyen, Madeleine Gibescu, and Antonio Liotta. Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science. *Nature communications*, 9(1):1–12, 2018.
- [51] A Martina Neuman and Jason J Bramburger. Transferability of graph neural networks using graphon and sampling theories. *arXiv preprint arXiv:2307.13206*, 2023.
- [52] Bithika Pal, Arindam Biswas, Sudeshna Kolay, Pabitra Mitra, and Biswajit Basu. A study on the ramanujan graph property of winning lottery tickets. In *International Conference on Machine Learning*, pages 17186–17201. PMLR, 2022.

- [53] Shreyas Malakarjun Patil and Constantine Dovrolis. Phew: Constructing sparse networks that learn fast and generalize well without training data. In *International Conference on Machine Learning*, pages 8432–8442. PMLR, 2021.
- [54] Ankit Pensia, Shashank Rajput, Alliot Nagle, Harit Vishwakarma, and Dimitris Papailiopoulos. Optimal lottery tickets via subset sum: Logarithmic over-parameterization is sufficient. *Advances in neural information processing systems*, 33:2599–2610, 2020.
- [55] Hoang Pham, Shiwei Liu, Lichuan Xiang, Dung Le, Hongkai Wen, Long Tran-Thanh, et al. Towards data-agnostic pruning at initialization: what makes a good sparse mask? *Advances in Neural Information Processing Systems*, 36:80044–80065, 2023.
- [56] Luana Ruiz, Luiz Chamon, and Alejandro Ribeiro. Graphon neural networks and the transferability of graph neural networks. *Advances in Neural Information Processing Systems*, 33:1702–1712, 2020.
- [57] Mahalakshmi Sabanayagam, Leena Chennuru Vankadara, and Debarghya Ghoshdastidar. Graphon based clustering and testing of networks: Algorithms and theory. *arXiv preprint arXiv:2110.02722*, 2021.
- [58] Yi Su, Raymond KW Wong, and Thomas CM Lee. Network estimation via graphon with node features. *IEEE Transactions on Network Science and Engineering*, 7(3):2078–2089, 2020.
- [59] Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. A simple and effective pruning approach for large language models. In *The Twelfth International Conference on Learning Representations*, 2024.
- [60] Hidenori Tanaka, Daniel Kunin, Daniel L Yamins, and Surya Ganguli. Pruning neural networks without any data by iteratively conserving synaptic flow. *Advances in Neural Information Processing Systems*, 33:6377–6389, 2020.
- [61] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- [62] Dharma Teja Vooturi, Girish Varma, and Kishore Kothapalli. Ramanujan bipartite graph products for efficient block sparse neural networks. *Concurrency and Computation: Practice and Experience*, 35(14):e6363, 2023.
- [63] Chaoqi Wang, Guodong Zhang, and Roger Grosse. Picking winning tickets before training by preserving gradient flow. In *International Conference on Learning Representations*, 2020.
- [64] Huan Wang, Can Qin, Yue Bai, Yulun Zhang, and Yun Fu. Recent advances on neural network pruning at initialization. In *International Joint Conference on Artificial Intelligence*, 2022.
- [65] Yite Wang, Dawei Li, and Ruoyu Sun. NTK-SAP: Improving neural network pruning by aligning training dynamics. In *The Eleventh International Conference on Learning Representations*, 2023.
- [66] Xinyue Xia, Gal Mishne, and Yusu Wang. Implicit graphon neural representation. In *International Conference on Artificial Intelligence and Statistics*, pages 10619–10634. PMLR, 2023.
- [67] Lichuan Xiang, Quan Nguyen-Tri, Lan-Cuong Nguyen, Hoang Pham, Khoat Than, Long Tran-Thanh, and Hongkai Wen. DPai: Differentiable pruning at initialization with node-path balance principle. In *The Thirteenth International Conference on Learning Representations*, 2025.
- [68] Hongteng Xu, Dixin Luo, Lawrence Carin, and Hongyuan Zha. Learning graphons via structured gromov-wasserstein barycenters. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10505–10513, 2021.
- [69] Greg Yang. Tensor programs ii: Neural tangent kernel for any architecture. *arXiv preprint arXiv:2006.14548*, 2020.

- 561 [70] Greg Yang and Etai Littwin. Tensor programs iib: Architectural universality of neural tangent
562 kernel training dynamics. In Marina Meila and Tong Zhang, editors, *Proceedings of the*
563 *38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine*
564 *Learning Research*, pages 11762–11772. PMLR, 18–24 Jul 2021.
- 565 [71] Hongru Yang, Ziyu Jiang, Ruizhe Zhang, Zhangyang Wang, and Yingbin Liang. Convergence
566 and generalization of wide neural networks with large bias. *arXiv preprint arXiv:2301.00327*,
567 2023.
- 568 [72] Hongru Yang and Zhangyang Wang. On the neural tangent kernel analysis of randomly pruned
569 neural networks. In Francisco Ruiz, Jennifer Dy, and Jan-Willem van de Meent, editors,
570 *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*,
571 volume 206 of *Proceedings of Machine Learning Research*, pages 1513–1553. PMLR, 25–27
572 Apr 2023.

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [\[Yes\]](#)

Justification: Our main claims are in the abstract and the listed contributions in Section 1 reflect the main contributions made in the paper.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [\[Yes\]](#)

Justification: We have provided the limitation discussion in Appendix E.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [\[Yes\]](#)

Justification: We have provided the necessary assumptions and proofs in the Appendix.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We have provided the information for reproducing the main experimental results and experimental settings in Sections 4, 6 and in the Appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We have provided the code in the supplemental material.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We have provided details in experiment settings in the Appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We have conducted tests for the statistical significance of the experiments.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [No]

Justification: We only provide the type of compute workers, but not the memory and time of execution.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: We make sure to conform to the NeurIPS Code of Ethics in every respect of the paper.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA]

Justification: This paper focuses on understanding the training dynamics of sparse neural networks, which is irrelevant to societal impacts.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: This paper focuses on understanding the training dynamics of sparse neural networks, which has no such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We have cited the original paper that produced the code package used in our paper.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset’s creators.

13. New assets

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: We did not release new assets

Guidelines: :

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. Crowdsourcing and research with human subjects

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. Institutional review board (IRB) approvals or equivalent for research with human subjects

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: The paper doesn’t involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

886 **16. Declaration of LLM usage**
887 Question: Does the paper describe the usage of LLMs if it is an important, original, or
888 non-standard component of the core methods in this research? Note that if the LLM is used
889 only for writing, editing, or formatting purposes and does not impact the core methodology,
890 scientific rigorousness, or originality of the research, declaration is not required.
891 Answer: [NA]
892 Justification: this paper does not involve LLMs as any important, original, or non-standard
893 components.
894 Guidelines:
895 • The answer NA means that the core method development in this research does not
896 involve LLMs as any important, original, or non-standard components.
897 • Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>)
898 for what should or should not be described.

899	Contents	
900	1 Introduction	1
901	2 Related work	3
902	3 Preliminaries	4
903	3.1 Neural tangent kernel	4
904	3.2 Graphon and graph limit theory	4
905	4 Graph limit and sparse neural networks	4
906	4.1 Pruning masks as graphs	5
907	4.2 Graphon Limit Hypothesis for neural network pruning	5
908	4.3 Experiments on graph limit of pruning at initialisation methods	5
909	5 Neural tangent kernel with graphon structure	6
910	5.1 Network structure and setup	6
911	5.2 Graphon neural tangent kernel	7
912	5.3 Graphon neural tangent kernel of Random Pruning	8
913	6 Numerical experiments	8
914	7 Conclusion	9
915	A Details on graphon neural tangent kernel	23
916	A.1 Network structure and setup	23
917	A.2 Forward propagation: covariance structure (proof of proposition 1)	25
918	A.3 Graphon neural tangent kernel convergence (proof of theorem 1)	27
919	B Details on graphon neural tangent kernel of Random pruning	30
920	C Experiments on graph limit of pruning at initialisation methods	32
921	D Details on numerical experiments	35
922	E Limitations and future research directions	36

A Details on graphon neural tangent kernel

In this section, we present a comprehensive derivation of the NTK for neural networks with graphon structure. This novel formulation extends the standard NTK theory to accommodate networks where connectivity patterns are modulated by graphon functions, providing insights into how network architecture influences learning dynamics. In particular, we first describe the network setting and the transition from discrete to continuous network in Appendix A.1. Then we provide the proof for the Proposition 1 in Appendix A.2 and for Theorem 1 in Appendix A.3, respectively.

A.1 Network structure and setup

Discrete neural network with graphon Consider a single output neural network with L hidden layers where weights are modulated by a graphon function $\mathcal{W}^{(l)} : [0, 1]^2 \rightarrow [0, 1]$:

$$W_{ij}^{(l)} \sim \mathcal{N}\left(0, \mathcal{W}^{(l)}\left(\frac{i}{n_l}, \frac{j}{n_{l-1}}\right)\right). \quad (5)$$

The graphon function governs the statistical structure and strength of synaptic connections between neurons. Specifically, it describes how the variance of random weights varies based on the positions of the connected neurons, thereby shaping the connectivity patterns and signal propagation across layers. Unlike in standard neural networks where weights are identically distributed, graphon-modulated weights have position-dependent variances while maintaining their independence.

Pre-activations and activations are computed as:

$$z_i^{(l)}(x) = \frac{1}{\sqrt{n_{l-1}}} \sum_{j=1}^{n_{l-1}} W_{ij}^{(l)} h_j^{(l-1)}(x), \quad (6)$$

$$h_i^{(l)}(x) = \sigma(z_i^{(l)}(x)). \quad (7)$$

The network output for simplicity is a single output:

$$f(x) = \frac{1}{\sqrt{n_L}} \sum_{j=1}^{n_L} W_{ij}^{(L+1)} h_j^{(L)}(x). \quad (8)$$

Connection to network pruning The graphon framework can be directly connected to network pruning. In standard pruning, a binary mask $M^{(l)}$ is applied to the weights:

$$\widetilde{W}^{(l)} = W^{(l)} \odot M^{(l)}, \quad (9)$$

where \odot denotes element-wise multiplication and $M_{ij}^{(l)} \in \{0, 1\}$ indicates whether the connection is kept (1) or pruned (0). In the finite-width case, the variance of the pruned weights becomes:

$$\text{Var}(\widetilde{W}_{ij}^{(l)}) = \sigma_w^2 M_{ij}^{(l)}, \quad (10)$$

which equals either σ_w^2 (kept) or 0 (pruned). As network width increases ($n_l, n_{l-1} \rightarrow \infty$), the pruning mask can be viewed as a step function over $[0, 1]^2$, with $M_{ij}^{(l)}$ corresponding to the value at $\left(\frac{i}{n_l}, \frac{j}{n_{l-1}}\right)$. In the limit, this mask converges to the graphon $\mathcal{W}^{(l)}(u_l, u_{l-1})$, which represents the density or probability of connections in each region of the unit square. Different pruning methods yield different graphon structures. Thus, the graphon-modulated weights can be interpreted as modeling the effect of pruning directly within the network initialization, enabling analysis of pruned network behavior in the infinite-width limit.

Continuous limit formulation As layer widths approach infinity ($n_l \rightarrow \infty$), we transition to continuous indices:

- $i/n_l \rightarrow u_l \in [0, 1]$ (position in layer l),

954 • $j/n_{l-1} \rightarrow u_{l-1} \in [0, 1]$ (position in layer $l - 1$).

955 The continuous network becomes:

$$z^{(l)}(u_l, x) = \int_0^1 W^{(l)}(u_l, u_{l-1}) h^{(l-1)}(u_{l-1}, x) du_{l-1}, \quad (11)$$

$$h^{(l)}(u_l, x) = \sigma(z^{(l)}(u_l, x)), \quad (12)$$

$$f(x) = \int_0^1 W^{(L+1)}(u_{L+1}, u_L) h^{(L)}(u_L, x) du_L, \quad (13)$$

956 where $W^{(l)}(u_l, u_{l-1}) \sim \mathcal{N}(0, \mathcal{W}^{(l)}(u_l, u_{l-1}))$.

957 **Statistical properties and convergence conditions** In graphon-structured networks, weights
 958 become independent but not identically distributed (non-i.i.d.) random variables. This change
 959 requires careful consideration of the conditions under which the Law of Large Numbers (LLN) and
 960 Central Limit Theorem (CLT) apply.

961 *Law of Large Numbers in Graphon Networks*

962 For the LLN to hold in the non-i.i.d. setting of graphon networks, the following conditions are
 963 required:

964 • **Bounded Graphon Values:** $\mathcal{W}^{(l)}(u_l, u_{l-1}) \leq 1$ ensures that $\text{Var}(W_{ij}^{(l)})$ is bounded, pre-
 965 venting any term from dominating the sum (satisfied by graphon definition).

966 • **Well-Defined Average Connectivity:** The integral $\int_0^1 \mathcal{W}^{(l)}(u_l, u_{l-1}) du_{l-1}$ must be well-
 967 defined for all positions u_l , ensuring the "average effect" of connections per neuron is
 968 stable.

969 Since the graphon values are bounded, the variances of weights and related quantities remain bounded,
 970 ensuring the LLN applies to empirical averages throughout the network.

971 *Central Limit Theorem and the Lindeberg-Feller Condition*

972 For pre-activations to converge to Gaussian processes in graphon networks, we assume the Lindeberg-
 973 Feller condition [6] is satisfied:

$$\lim_{n_{l-1} \rightarrow \infty} \frac{1}{\sigma_n^2} \sum_{j=1}^{n_{l-1}} \mathbb{E} [X_j^2 \cdot \mathbf{1}_{\{|z_j| > \epsilon \sigma_n\}}] = 0, \quad (14)$$

974 for all $\epsilon > 0$, where $X_j = \frac{1}{\sqrt{n_{l-1}}} W_{ij}^{(l)} h_j^{(l-1)}(x)$, $\sigma_n^2 = \sum_{j=1}^{n_{l-1}} \text{Var}(X_j)$, and the pre-activation
 975 $z_i = \sum_j^{n_{l-1}} X_j$. This condition ensures that no single term in the sum disproportionately influences
 976 the total variance. Then, the CLT applies despite the non-i.i.d. nature of the weights, allowing
 977 pre-activations to converge to Gaussian processes with position-dependent covariance structures.

978 In our setting, the Lindeberg condition is satisfied due to the following structural properties of graphon
 979 networks: (i) When network weights are initialized using a bounded graphon ($\mathcal{W}^{(l)}(u, v) \leq 1$), each
 980 connection's variance is controlled, preventing any single weight from becoming dominant; (ii)
 981 Activations from the previous layer $h_j^{(l-1)}(x)$ are bounded with sigmoid and tanh function, while
 982 activations like ReLU tend to remain within reasonable ranges when networks are properly initialized
 983 and inputs are normalized; (iii) The scaling factor of $\frac{1}{\sqrt{n_{l-1}}}$ in the pre-activation formula ensures
 984 that individual pre-activation variances decrease proportionally as the network widens, scaling as
 985 $O(\frac{1}{n_{l-1}})$. Together, these properties imply that as the layer width $n_{l-1} \rightarrow \infty$, the influence of any
 986 single term X_j becomes negligible relative to the total variance. Hence, the Lindeberg condition is
 987 met, and the pre-activations converge in distribution to a Gaussian process with a position-dependent
 988 covariance structure induced by the graphon.

989 A.2 Forward propagation: covariance structure (proof of proposition 1)

990 We first establish the statistical behaviour of signals as they propagate through the network. Our main
 991 result for the forward pass is captured in the Proposition 1. For the sake of convenience, we re-state
 992 the proposition here again:

993 **Proposition 2.** *For a neural network with layers structured by graphons $\mathcal{W}^{(l)} : [0, 1]^2 \rightarrow [0, 1]$,
 994 Lipschitz nonlinearity σ , and in the limit as $n_1, \dots, n_L \rightarrow \infty$, the pre-activations $z^{(l)}(u_l, x)$ at every
 995 hidden layer converge to centred Gaussian processes with covariance $\tilde{\Sigma}^{(l)}$, where $\tilde{\Sigma}^{(l)}$ is defined
 996 recursively by:*

$$\tilde{\Sigma}^{(1)}(u_1, u'_1, x, x') = \delta(u_1 - u'_1) \frac{1}{d} \sum_j^d \mathcal{W}^{(1)}(u_1, \frac{j}{d})(x \cdot x')_j, \quad (15)$$

$$\tilde{\Sigma}^{(l)}(u_l, u'_l, x, x') = \delta(u_l - u'_l) \int_0^1 \mathcal{W}^{(l)}(u_l, u_{l-1}) \Sigma^{(l-1)}(u_{l-1}, u_{l-1}, x, x') du_{l-1}, \quad (16)$$

997 where $(x \cdot x')_j$ represents the input correlation at position j , and the activation covariance $\Sigma^{(l)}$ is:

$$\Sigma^{(l)}(u_l, u'_l, x, x') = \delta(u_l - u'_l) \mathbb{E}_{(z, z') \sim \mathcal{N}(0, \Lambda^{(l)}(u_l))} [\sigma(z) \sigma(z')], \quad (17)$$

998 where $\delta(u_l - u'_l)$ is Dirac delta function, $\Lambda^{(l)}(u_l)$ is the position-dependent covariance matrix:

$$\Lambda^{(l)}(u_l) = \begin{bmatrix} \tilde{\Sigma}^{(l)}(u_l, u_l, x, x) & \tilde{\Sigma}^{(l)}(u_l, u_l, x, x') \\ \tilde{\Sigma}^{(l)}(u_l, u_l, x', x) & \tilde{\Sigma}^{(l)}(u_l, u_l, x', x') \end{bmatrix}.$$

1000 *Proof:*

1001 We prove the proposition by induction on the layer index l . The key insight is to analyze how the
 1002 graphon structure affects the statistical properties of pre-activations as signals propagate through the
 1003 network. The proof works as follows:

1004 **Base case: first layer $l = 1$** For the first layer with graphon modulation:

$$W_{ij}^{(1)} \sim \mathcal{N}\left(0, \mathcal{W}^{(1)}\left(\frac{i}{n_1}, \frac{j}{d}\right)\right), \quad (18)$$

1005 where d is the input dimension. The pre-activation covariance becomes:

$$\mathbb{E}[z_i^{(1)}(x) z_{i'}^{(1)}(x')] = \mathbb{E}\left[\left(\frac{1}{\sqrt{d}} \sum_{j=1}^d W_{ij}^{(1)} x_j\right) \left(\frac{1}{\sqrt{d}} \sum_{k=1}^d W_{i'k}^{(1)} x'_k\right)\right] \quad (19)$$

$$= \frac{1}{d} \sum_{j=1}^d \sum_{k=1}^d \mathbb{E}[W_{ij}^{(1)} W_{i'k}^{(1)}] x_j x'_k. \quad (20)$$

1006 Since weights are independently sampled:

- 1007 • $\mathbb{E}[W_{ij}^{(1)} W_{i'k}^{(1)}] = 0$ when $(i, j) \neq (i', k)$,
- 1008 • $\mathbb{E}[(W_{ij}^{(1)})^2] = \mathcal{W}^{(1)}\left(\frac{i}{n_1}, \frac{j}{d}\right)$ when $i = i'$ and $j = k$.

1009 Therefore:

$$\mathbb{E}[z_i^{(1)}(x) z_{i'}^{(1)}(x')] = \delta_{ii'} \frac{1}{d} \left[\sum_{j=1}^d \mathcal{W}^{(1)}\left(\frac{i}{n_1}, \frac{j}{d}\right) x_j x'_j \right]. \quad (21)$$

1010 As $n_1 \rightarrow \infty$ in the continuous limit with $i/n_1 \rightarrow u_1$:

$$\tilde{\Sigma}^{(1)}(u_1, u'_1, x, x') = \delta(u_1 - u'_1) \frac{1}{d} \sum_j^d \mathcal{W}^{(1)}(u_1, \frac{j}{d})(x \cdot x')_j, \quad (22)$$

1011 where \cdot is the dot-product, $(x \cdot x')_j$ represents the input correlation at position j .

1012 The activation covariance at position u_1 is:

$$\Sigma^{(1)}(u_1, u'_1, x, x') = \delta(u_1 - u'_1) \mathbb{E}[\sigma(z^{(1)}(u_1, x))\sigma(z^{(1)}(u_1, x'))]. \quad (23)$$

1013 Since $(z^{(1)}(u_1, x), z^{(1)}(u_1, x'))$ follows a joint Gaussian distribution according to the CLT, this can
1014 be computed as:

$$\Sigma^{(1)}(u_1, u'_1, x, x') = \delta(u_1 - u'_1) \mathbb{E}_{(z, z') \sim \mathcal{N}(0, \Lambda^{(1)}(u_1))}[\sigma(z)\sigma(z')] \quad (24)$$

1015 Where $\Lambda^{(1)}(u_1)$ is the position-dependent covariance matrix:

$$\Lambda^{(1)}(u_1) = \begin{bmatrix} \tilde{\Sigma}^{(1)}(u_1, u_1, x, x) & \tilde{\Sigma}^{(1)}(u_1, u_1, x, x') \\ \tilde{\Sigma}^{(1)}(u_1, u_1, x', x) & \tilde{\Sigma}^{(1)}(u_1, u_1, x', x') \end{bmatrix}. \quad (25)$$

1016 Different from fully connected network, where the pre-activation covariance of the first layer becomes
1017 covariance of the input layer $\Sigma^{(0)}(x, x') = x \cdot x'$, in graphon networks, the pre-activation covariance
1018 depends on the structure of the graphon.

1019 **Inductive step: subsequent layer $l > 1$** For layer l with graphon structure:

$$\mathbb{E}[z_i^{(l)}(x)z_{i'}^{(l)}(x')] = \mathbb{E}\left[\frac{1}{n_{l-1}} \left(\sum_{j=1}^{n_{l-1}} W_{ij}^{(l)} h_j^{(l-1)}(x)\right) \left(\sum_{k=1}^{n_{l-1}} W_{i'k}^{(l)} h_k^{(l-1)}(x')\right)\right] \quad (26)$$

$$= \frac{1}{n_{l-1}} \sum_{j=1}^{n_{l-1}} \sum_{k=1}^{n_{l-1}} \mathbb{E}[W_{ij}^{(l)} W_{i'k}^{(l)}] \mathbb{E}[h_j^{(l-1)}(x) h_k^{(l-1)}(x')]. \quad (27)$$

1020 Since weights are independently sampled and $\mathbb{E}[(W_{ij}^{(l)})^2] = \mathcal{W}^{(l)}\left(\frac{i}{n_l}, \frac{j}{n_{l-1}}\right)$:

$$\mathbb{E}[z_i^{(l)}(x)z_{i'}^{(l)}(x')] = \delta_{ii'} \left[\frac{1}{n_{l-1}} \sum_{j=1}^{n_{l-1}} \mathcal{W}^{(l)}\left(\frac{i}{n_l}, \frac{j}{n_{l-1}}\right) \mathbb{E}[h_j^{(l-1)}(x) h_j^{(l-1)}(x')] \right]. \quad (28)$$

1021 In the continuous limit as $j/n_{l-1} \rightarrow u_{l-1}$, and $1/n_{l-1}$ is absorbed into the integral:

$$\tilde{\Sigma}^{(l)}(u_l, u'_l, x, x') = \delta(u_l - u'_l) \int_0^1 \mathcal{W}^{(l)}(u_l, u_{l-1}) \Sigma^{(l-1)}(u_{l-1}, u_{l-1}, x, x') du_{l-1}. \quad (29)$$

1022 The activation covariance is:

$$\Sigma^{(l)}(u_l, u'_l, x, x') = \delta(u_l - u'_l) \mathbb{E}_{(z, z') \sim \mathcal{N}(0, \Lambda^{(l)}(u_l))}[\sigma(z)\sigma(z')], \quad (30)$$

1023 where $\Lambda^{(l)}(u_l)$ is the covariance matrix for pre-activations at position u_l :

$$\Lambda^{(l)}(u_l) = \begin{bmatrix} \tilde{\Sigma}^{(l)}(u_l, u_l, x, x) & \tilde{\Sigma}^{(l)}(u_l, u_l, x, x') \\ \tilde{\Sigma}^{(l)}(u_l, u_l, x', x) & \tilde{\Sigma}^{(l)}(u_l, u_l, x', x') \end{bmatrix}. \quad (31)$$

1024 This completes the proof of Proposition 1 by induction. A critical insight is how the graphon structure
1025 $\mathcal{W}^{(l)}(u_l, u_{l-1})$ directly modulates signal propagation, creating non-uniform information flow across
1026 different network regions.

1027 A.3 Graphon neural tangent kernel convergence (proof of theorem 1)

1028 Our main theoretical result characterises the NTK for graphon-structured networks. For convenience,
1029 we re-state the Theorem 1 here again:

1030 **Theorem 2** (Graphon NTK). *For a neural network with layers structured by graphons $\mathcal{W}^{(l)} : [0, 1]^2 \rightarrow [0, 1]$, Lipschitz nonlinearity σ , in the limit as $n_1, \dots, n_L \rightarrow \infty$, the Graphon Neural
1031 Tangent Kernel (Graphon NTK) $\Theta(x, x')$ converges to a deterministic kernel:*

$$\Theta(x, x') = \sum_{l=1}^L \int_0^1 \left(\dot{\Sigma}^{(l)}(u_l, u_l, x, x') \int_{[0,1]^{L-l+1}} \prod_{m=l+1}^{L+1} \mathcal{W}^{(m)}(u_m, u_{m-1}) \dot{\Sigma}^{(m)}(u_m, u_m, x, x') d\mathbf{u}_{l+1} \right) \left(\int_0^1 \Sigma^{(l-1)}(u_{l-1}, u_{l-1}, x, x') du_{l-1} \right) du_l \quad (32)$$

1033 where $\dot{\Sigma}^{(l)}(u_l, u_l, x, x') = \mathbb{E}[\sigma'(z^{(l)}(u_l, x))\sigma'(z^{(l)}(u_l, x'))]$ represents the expected correlation
1034 between activation derivatives, and $d\mathbf{u}_{l+1} = du_{L+1} du_L \dots du_{l+1}$.

1035 *Proof:*

1036 We prove the Graphon NTK by decomposing the derivation into three key steps: (1) characterizing
1037 gradient flow through graphon-structured layers, (2) analyzing gradient correlations under infinite-
1038 width statistical independence, and (3) integrating these correlations to derive a closed-form NTK
1039 expression. This structured approach yields an interpretable kernel that reflects how the graphon
1040 structure shapes learning dynamics across the network.

1041 A.3.1 Backward propagation: gradient flow

1042 **Gradient recursion** For layer L , applying the chain rule:

$$\frac{\partial f(x)}{\partial z^{(L)}(u_L, x)} = \frac{\partial f(x)}{\partial h^{(L)}(u_L, x)} \frac{\partial h^{(L)}(u_L, x)}{\partial z^{(L)}(u_L, x)} = W^{(L+1)}(u_{L+1}, u_L) \sigma'(z^{(L)}(u_L, x)). \quad (33)$$

1043 For earlier layers ($l < L$), the gradient with respect to pre-activations is:

$$\frac{\partial f(x)}{\partial z^{(l)}(u_l, x)} = \sigma'(z^{(l)}(u_l, x)) \int_0^1 \frac{\partial f(x)}{\partial z^{(l+1)}(u_{l+1}, x)} W^{(l+1)}(u_{l+1}, u_l) du_{l+1}. \quad (34)$$

1044 **Parameter gradients** The gradients with respect to weights are:

$$\frac{\partial f(x)}{\partial W^{(l)}(u_l, u_{l-1})} = \frac{\partial f(x)}{\partial z^{(l)}(u_l, x)} h^{(l-1)}(u_{l-1}, x). \quad (35)$$

1045 A.3.2 Graphon neural tangent kernel derivation

1046 The Neural Tangent Kernel is defined as the inner product of gradients with respect to all parameters:

$$\Theta(x, x') = \sum_{l=1}^L \int_0^1 \int_0^1 \mathbb{E} \left[\frac{\partial f(x)}{\partial W^{(l)}(u_l, u_{l-1})} \frac{\partial f(x')}{\partial W^{(l)}(u_l, u_{l-1})} \right] du_l du_{l-1}. \quad (36)$$

1047 **Layer-wise graphon NTK contribution** The contribution to the Graphon NTK from layer l is:

$$\Theta_l(x, x') = \int_0^1 \int_0^1 \mathbb{E} \left[\frac{\partial f(x)}{\partial W^{(l)}(u_l, u_{l-1})} \frac{\partial f(x')}{\partial W^{(l)}(u_l, u_{l-1})} \right] du_l du_{l-1} \quad (37)$$

$$= \int_0^1 \int_0^1 \mathbb{E} \left[\frac{\partial f(x)}{\partial z^{(l)}(u_l, x)} h^{(l-1)}(u_{l-1}, x) \frac{\partial f(x')}{\partial z^{(l)}(u_l, x')} h^{(l-1)}(u_{l-1}, x') \right] du_l du_{l-1}. \quad (38)$$

At this point, we apply a key insight: In the infinite-width limit, by the LLN, the gradients $\frac{\partial f(x)}{\partial z^{(l)}(u_l, x)}$ and the activations $h^{(l-1)}(u_{l-1}, x)$ become statistically independent for distinct positions u_l and u_{l-1} . This allows us to factorize the expectation:

$$\Theta_l(x, x') = \int_0^1 \mathbb{E} \left[\frac{\partial f(x)}{\partial z^{(l)}(u_l, x)} \frac{\partial f(x')}{\partial z^{(l)}(u_l, x')} \right] \left(\int_0^1 \mathbb{E}[h^{(l-1)}(u_{l-1}, x) h^{(l-1)}(u_{l-1}, x')] du_{l-1} \right) du_l \quad (39)$$

$$= \int_0^1 \mathbb{E} \left[\frac{\partial f(x)}{\partial z^{(l)}(u_l, x)} \frac{\partial f(x')}{\partial z^{(l)}(u_l, x')} \right] \left(\int_0^1 \Sigma^{(l-1)}(u_{l-1}, u_{l-1}, x, x') du_{l-1} \right) du_l. \quad (40)$$

Gradient correlation analysis To compute the NTK, we need to analyze the correlation between gradients at different inputs:

$$\mathbb{E} \left[\frac{\partial f(x)}{\partial z^{(l)}(u_l, x)} \frac{\partial f(x')}{\partial z^{(l)}(u_l, x')} \right]. \quad (41)$$

From our backward propagation analysis in Equation 34, we have:

$$\begin{aligned} \mathbb{E} \left[\frac{\partial f(x)}{\partial z^{(l)}(u_l, x)} \frac{\partial f(x')}{\partial z^{(l)}(u_l, x')} \right] &= \mathbb{E} \left[\sigma'(z^{(l)}(u_l, x)) \sigma'(z^{(l)}(u_l, x')) \right. \\ &\quad \left. \int_0^1 \int_0^1 \frac{\partial f(x)}{\partial z^{(l+1)}(u_{l+1}, x)} \frac{\partial f(x')}{\partial z^{(l+1)}(u'_{l+1}, x')} W^{(l+1)}(u_{l+1}, u_l) W^{(l+1)}(u'_{l+1}, u_l) du_{l+1} du'_{l+1} \right]. \end{aligned} \quad (42)$$

In the infinite-width limit, $(z^{(l)}(u_l, x), z^{(l)}(u_l, x'))$ follows a bivariate Gaussian distribution, allowing us to define:

$$\dot{\Sigma}^{(l)}(u_l, u_l, x, x') = \mathbb{E}[\sigma'(z^{(l)}(u_l, x)) \sigma'(z^{(l)}(u_l, x'))]. \quad (43)$$

The weights $W^{(l+1)}(u_{l+1}, u_l)$ and $W^{(l+1)}(u'_{l+1}, u_l)$ are independent for $u_{l+1} \neq u'_{l+1}$, with:

$$\mathbb{E}[W^{(l+1)}(u_{l+1}, u_l) W^{(l+1)}(u'_{l+1}, u_l)] = \mathcal{W}^{(l+1)}(u_{l+1}, u_l) \delta(u_{l+1} - u'_{l+1}). \quad (44)$$

Substituting this into our gradient correlation:

$$\begin{aligned} \mathbb{E} \left[\frac{\partial f(x)}{\partial z^{(l)}(u_l, x)} \frac{\partial f(x')}{\partial z^{(l)}(u_l, x')} \right] &= \dot{\Sigma}^{(l)}(u_l, u_l, x, x') \\ &\quad \int_0^1 \mathcal{W}^{(l+1)}(u_{l+1}, u_l) \mathbb{E} \left[\frac{\partial f(x)}{\partial z^{(l+1)}(u_{l+1}, x)} \frac{\partial f(x')}{\partial z^{(l+1)}(u_{l+1}, x')} \right] du_{l+1}. \end{aligned} \quad (45)$$

Closed-form expression for Graphon NTK To derive the closed-form expression, we define:

$$G^{(l)}(u_l, x, x') = \mathbb{E} \left[\frac{\partial f(x)}{\partial z^{(l)}(u_l, x)} \frac{\partial f(x')}{\partial z^{(l)}(u_l, x')} \right]. \quad (46)$$

From our derivation, $G^{(l)}$ follows the recursion:

$$G^{(l)}(u_l, x, x') = \dot{\Sigma}^{(l)}(u_l, u_l, x, x') \int_0^1 \mathcal{W}^{(l+1)}(u_{l+1}, u_l) G^{(l+1)}(u_{l+1}, x, x') du_{l+1}. \quad (47)$$

With the base case $G^{(L+1)}(u_{L+1}, x, x') = 1$.

1061 The general form for any layer l becomes:

$$G^{(l)}(u_l, x, x') = \dot{\Sigma}^{(l)}(u_l, u_l, x, x') \int_{[0,1]^{L-l+1}} \prod_{m=l+1}^{L+1} \mathcal{W}^{(m)}(u_m, u_{m-1}) \dot{\Sigma}^{(m)}(u_m, u_m, x, x') d\mathbf{u}_{l+1}, \quad (48)$$

1062 where $d\mathbf{u}_{l+1} = du_{L+1} du_L \dots du_{l+1}$ and $\dot{\Sigma}^{(L+1)}(u_{L+1}, u_{L+1}, x, x') = 1$.

1063 Substituting this into our formula for Θ_l :

$$\Theta_l(x, x') = \int_0^1 G^{(l)}(u_l, x, x') \left(\int_0^1 \Sigma^{(l-1)}(u_{l-1}, u_{l-1}, x, x') du_{l-1} \right) du_l \quad (49)$$

$$= \int_0^1 \left(\dot{\Sigma}^{(l)}(u_l, u_l, x, x') \int_{[0,1]^{L-l+1}} \prod_{m=l+1}^{L+1} \mathcal{W}^{(m)}(u_m, u_{m-1}) \dot{\Sigma}^{(m)}(u_m, u_m, x, x') d\mathbf{u}_{l+1} \right) \quad (50)$$

$$\left(\int_0^1 \Sigma^{(l-1)}(u_{l-1}, u_{l-1}, x, x') du_{l-1} \right) du_l. \quad (51)$$

1064 The full Graphon NTK is the sum over all layers:

$$\Theta(x, x') = \sum_{l=1}^L \Theta_l(x, x') \quad (52)$$

$$= \sum_{l=1}^L \int_0^1 \left(\dot{\Sigma}^{(l)}(u_l, u_l, x, x') \int_{[0,1]^{L-l+1}} \prod_{m=l+1}^{L+1} \mathcal{W}^{(m)}(u_m, u_{m-1}) \dot{\Sigma}^{(m)}(u_m, u_m, x, x') d\mathbf{u}_{l+1} \right) \quad (53)$$

$$\left(\int_0^1 \Sigma^{(l-1)}(u_{l-1}, u_{l-1}, x, x') du_{l-1} \right) du_l \quad (54)$$

1065 This expression reveals how the graphon structure at each layer shapes the Neural Tangent Kernel
1066 through multiple integrals involving the graphon functions. The resulting kernel is position-dependent
1067 and reflects the specific connectivity patterns encoded by the graphons.

1068 A.3.3 Discussion and implication

1069 The Graphon NTK provides a powerful analytical framework for understanding how structured
1070 weight patterns influence neural network learning dynamics. Several key insights emerge:

- 1071 1. **Non-uniform signal propagation:** The graphon structure creates position-dependent infor-
1072 mation flow, with regions of higher graphon values propagating signals more strongly.
- 1073 2. **Relationship to pruning:** The graphon formulation provides a continuous limit perspective
1074 on network pruning, where the graphon $\mathcal{W}^{(l)}(u_l, u_{l-1})$ can be interpreted as the density or
1075 probability of connections.
- 1076 3. **Position-dependent learning dynamics:** Different regions of the network effectively learn
1077 at different rates based on their connectivity patterns.

1078 These theoretical results establish the foundation for analysing learning behaviours in neural networks
1079 with connectivity patterns, providing insights that may guide the development of more efficient
1080 architectural designs.

1081 B Details on graphon neural tangent kernel of Random pruning

1082 With Random pruning, pruning masks converge to constant graphons (the Erdős–Rényi random
1083 graph) as the width tends to infinity. When the underlying graphons are constants, we observe a
1084 uniform scaling effect on training dynamics.

1085 For a constant graphon $W(u, v) = c$, the resulting NTK scales uniformly as:

$$\Theta(x, x') = c^L \Theta_{\text{std}}(x, x'), \quad (55)$$

1086 where $\Theta_{\text{std}}^{(L)}$ denotes the standard NTK of a fully-connected network.

1087 *Proof:*

1088 For the first hidden layer ($l = 1$), pre-activation covariance becomes:

$$\tilde{\Sigma}^{(1)}(u, u', x, x') = \delta(u - u') (c(x \cdot x')). \quad (56)$$

1089 For simplicity, assuming $\sigma_b^2 = 0$ (no biases):

$$\tilde{\Sigma}^{(1)}(u, u', x, x') = c \tilde{\Sigma}_{\text{std}}^{(1)}(u, u', x, x'). \quad (57)$$

1090 This leads to activation covariance at first layer:

$$\Sigma^{(1)}(u, u', x, x') = c \Sigma_{\text{std}}^{(1)}(u, u', x, x'). \quad (58)$$

1091 By induction, for any layer l :

$$\begin{aligned} \tilde{\Sigma}^{(l)}(u, u', x, x') &= \delta(u - u') \left(c \int_0^1 \Sigma^{(l-1)}(v, v, x, x') dv \right) \\ &= \delta(u - u') \left(c \int_0^1 c^{l-1} \Sigma_{\text{std}}^{(l-1)}(v, v, x, x') dv \right) \\ &= \delta(u - u') \left(c^l \int_0^1 \Sigma_{\text{std}}^{(l-1)}(v, v, x, x') dv \right) \\ &= c^l \tilde{\Sigma}_{\text{std}}^{(l)}(u, u', x, x'). \end{aligned} \quad (59)$$

1092 And the activation covariance at layer l is:

$$\Sigma^{(l)}(u, u', x, x') = c^l \Sigma_{\text{std}}^{(l)}(u, u', x, x'). \quad (60)$$

1093 We also assume that the activation function is ReLU, then the relationship between activation
1094 derivative covariance:

$$\dot{\Sigma}^{(l)}(x, x') = \dot{\Sigma}_{\text{std}}^{(l)}(x, x'). \quad (61)$$

1095 For the gradient correlation function:

$$G^{(l)}(u, x, x') = \dot{\Sigma}^{(l)}(u, u, x, x') \int_0^1 \mathcal{W}^{(l+1)}(v, u) G^{(l+1)}(v, x, x') dv. \quad (62)$$

1096 For a homogeneous graphon, each backward step contributes a factor of c . Going from output layer
1097 $L + 1$ back to layer l :

$$\begin{aligned}
G^{(L+1)}(u, x, x') &= G_{\text{std}}^{(L+1)}(u, x, x') = 1 \\
G^{(L)}(u, x, x') &= \dot{\Sigma}^{(L)}(u, u, x, x') \int_0^1 \mathcal{W}^{(L+1)}(v, u) G^{(L+1)}(v, x, x') dv \\
G^{(L)}(u, x, x') &= c \dot{\Sigma}^{(L)}(u, u, x, x') \\
G^{(L)}(u, x, x') &= c G_{\text{std}}^{(L)}(u, x, x') \\
G^{(L-1)}(u, x, x') &= \dot{\Sigma}^{(L-1)}(u, u, x, x') c \int_0^1 G^{(L)}(v, x, x') dv \\
G^{(L-1)}(u, x, x') &= c^2 G_{\text{std}}^{(L-1)}(u, x, x')
\end{aligned}$$

1098 By induction, we have:

$$G^{(l)}(u, x, x') = c^{L+1-l} G_{\text{std}}^{(l)}(u, x, x'). \quad (63)$$

1099 The layer-wise NTK contribution is:

$$\Theta_l(x, x') = \int_0^1 G^{(l)}(u, x, x') \left(\int_0^1 \Sigma^{(l-1)}(v, v, x, x') dv \right) du. \quad (64)$$

1100 Substituting our findings:

$$\Theta_l(x, x') = \int_0^1 c^{L+1-l} \dot{\Sigma}^{(l)}(u, x, x') \left(\int_0^1 c^{l-1} \Sigma_{\text{std}}^{(l-1)}(v, v', x, x') dv \right) du. \quad (65)$$

1101 When the activation integral dominates the constant 1 (which is typical in deep networks):

$$\Theta_l(x, x') = c^{L+1-l} c^{l-1} \Theta_{l,\text{std}}(x, x') = c^L \Theta_{l,\text{std}}(x, x'). \quad (66)$$

1102 Summing over all layers:

$$\Theta(x, x') = \sum_{l=1}^L \Theta_l(x, x') = c^L \sum_{l=1}^L \Theta_{l,\text{std}}(x, x') = c^L \Theta_{\text{std}}(x, x'). \quad (67)$$

1103 This uniform scaling directly impacts convergence rates during training. If λ_k are the eigenvalues of
1104 Θ_{std} , then the eigenvalues of Θ become $c^L \lambda_k$. Consequently, residual modes during training decay
1105 as:

$$a_k(t) = a_k(0) e^{-c^L \lambda_k t}. \quad (68)$$

1106 This scaling directly influences network training dynamics. If λ_k are the eigenvalues of Θ_{std} , then
1107 the eigenvalues of the pruned network's NTK become $c^L \lambda_k$. Notably, while the absolute learning
1108 speed is reduced, the relative dynamics between modes remain unchanged. This offers a principled
1109 explanation for the empirical observation that sparse random networks converge more slowly than
1110 their dense counterparts [26].

1111 More broadly, our framework extends beyond this homogeneous case, allowing for arbitrary position-
1112 dependent graphons that induce heterogeneous learning dynamics. This highlights the generality and
1113 flexibility of Graphon NTK in modeling structured sparsity and its effect on neural dynamics.

1114 C Experiments on graph limit of pruning at initialisation methods

1115 We empirically validate the graphon hypothesis by examining whether pruning methods converge to
 1116 distinct, characteristic graphons as network width increases. We analyze four pruning-at-initialization
 1117 methods: Random pruning, SNIP [42], GraSP [63], and Synflow [60] across varying network widths
 1118 $n \in \{100, 500, 1000, 2000\}$, layers 4, 5, and sparsity levels $\{70\%, 80\%, 90\%\}$. We conduct 100
 1119 independent trials per configuration and exclude masks from input and output layers. We follow
 1120 Synflow [60] source code to produce masks ². All the experiments are run on a single Nvidia A10
 1121 GPU (24GB).

1122 To visualize the emergent graphons, we employ the SAS method [12] that:

- 1123 1. Sorts nodes based on degree centrality (out-degree for layer l , in-degree for layer $l + 1$). For
 1124 2D histogram, nodes in layer l and $l + 1$ are in the x-axis and y-axis, respectively.
- 1125 2. Partitions the sorted bipartite graph into a grid of intervals. Each axis is split into $K = 64$
 1126 intervals.
- 1127 3. Computes the average edge density within each interval.

1128 This degree-based sorting serves as an approximate measure-preserving transformation, revealing
 1129 underlying structural patterns while maintaining invariance to node permutations.

1130 We visualise graph limits of subnetworks' masks produced by PaI methods at different sparsity levels
 1131 in 4- and 5-layer networks in Figures 5, 6, 7, 8, and 9. In particular, Random pruning converges
 1132 to a constant graphon (Erdős-Rényi random graph), with uniform connection probability across all
 1133 node positions. SNIP and GraSP exhibit structured, non-uniform graphons with density gradients,
 1134 preferentially connecting high-centrality nodes. Synflow converges to a block-like graphon with
 1135 sharp transitions, strongly prioritising connections among high-centrality neurons. In Figure 9, a large
 1136 part of neurons is eliminated in hidden layers creating a subnetwork with high paths. This observation
 1137 is also indicated in [55, 53], in which after each iteration, Synflow prunes weights connected to
 1138 low-degree nodes.

1139 To quantify convergence, we show the Euclidean distance between density matrices at width N and
 1140 reference matrices at $N = 2000$ as in Figure 4. Since all histograms are aligned via degree-based
 1141 sorting, we used the Euclidean distance between the density matrices as a proxy for the cut distance

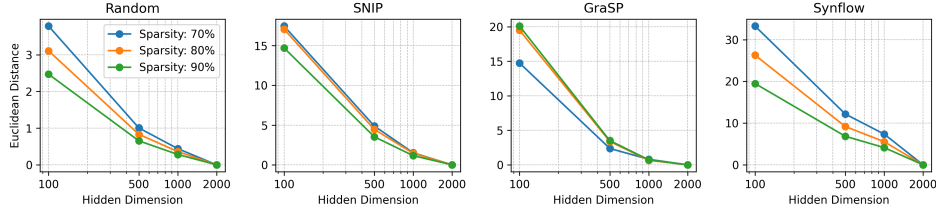
$$d(W_N, W_{2000}) = \sqrt{\sum_{i,j} (H_N(i, j) - H_{2000}(i, j))^2}.$$

1142 where H is the histogram.

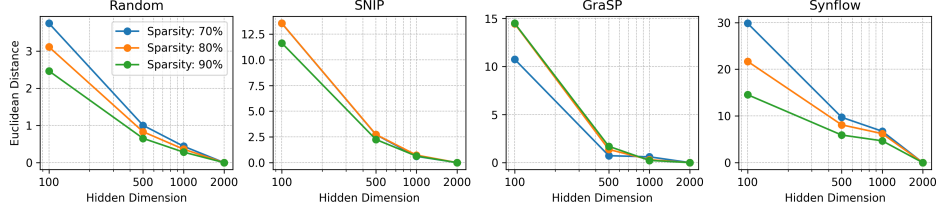
1143 All methods demonstrate monotonic convergence, with distances decreasing as width increases,
 1144 confirming that the limiting graphon structure is an intrinsic characteristic of each pruning algorithm.

1145 These results provide compelling evidence that each pruning method induces a unique (up to rela-
 1146 belling), stable connectivity pattern in the large-width limit, validating our graphon hypothesis and
 1147 establishing a foundation for analysing pruning methods through graph limit theory.

²<https://github.com/ganguli-lab/Synaptic-Flow>

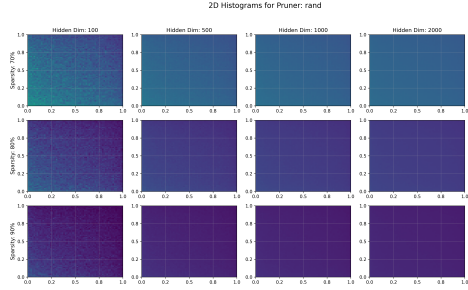


(a) Histogram convergence via Euclidean distance in 4-layer networks setting.

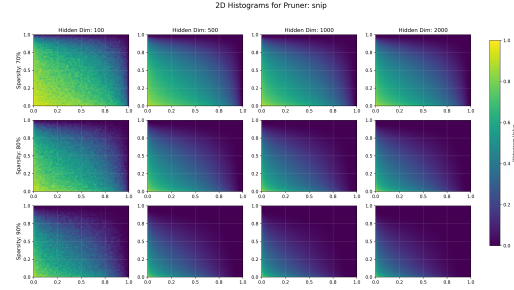


(b) Histogram convergence via Euclidean distance in 5-layer networks setting.

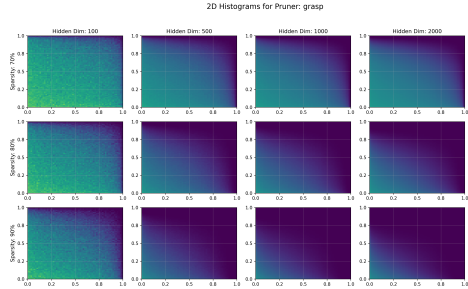
Figure 4: Histogram convergence via Euclidean distance.



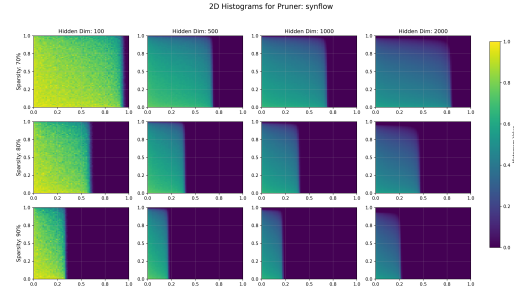
(a) Graph limit of subnetworks' mask produced by Random pruning at different sparsity levels in 4-layer networks.



(b) Graph limit of subnetworks' mask produced by SNIP pruning at different sparsity levels in 4-layer networks.



(c) Graph limit of subnetworks' mask produced by GraSP pruning at different sparsity levels in 4-layer networks.



(d) Graph limit of subnetworks' mask produced by Synflow pruning at different sparsity levels in 4-layer networks.

Figure 5: Graph limit of subnetworks' mask produced by PaI methods at different sparsity levels in 4-layer networks.

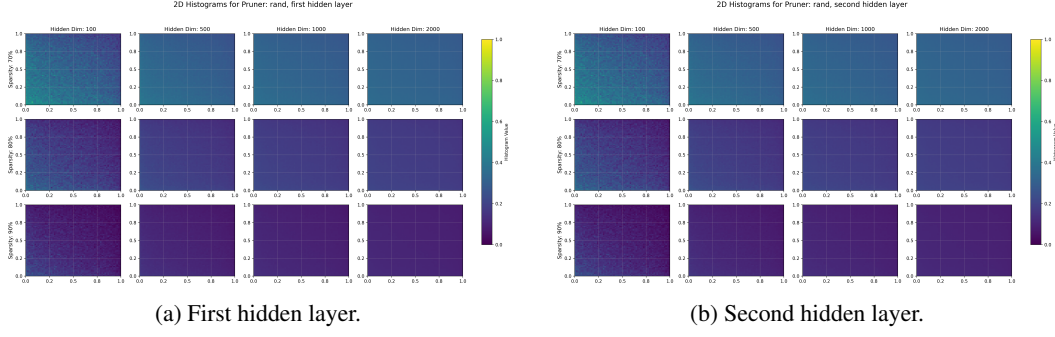


Figure 6: Graph limit of subnetworks' mask produced by Random pruning at different sparsity levels in 5-layer networks.

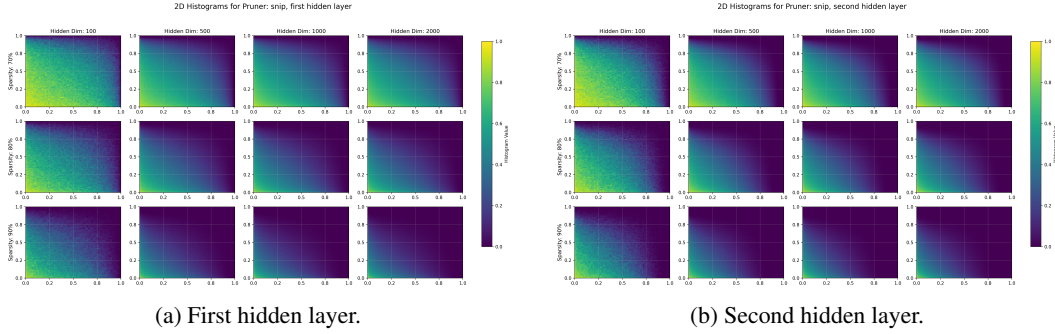


Figure 7: Graph limit of subnetworks' mask produced by SNIP pruning at different sparsity levels in 5-layer networks.

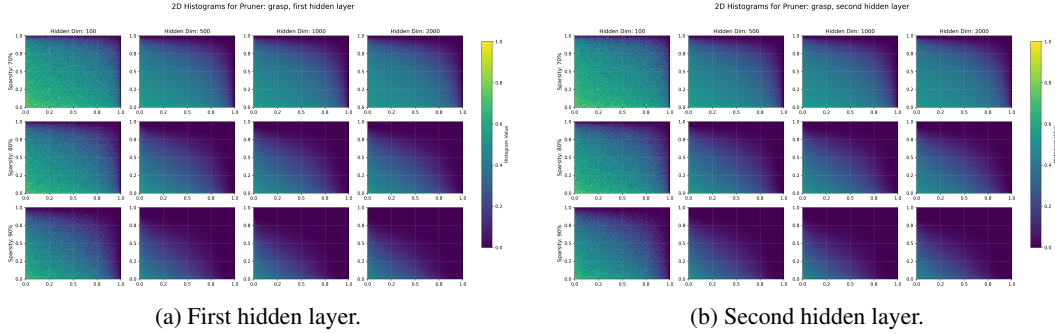


Figure 8: Graph limit of subnetworks' mask produced by GraSP pruning at different sparsity levels in 5-layer networks.

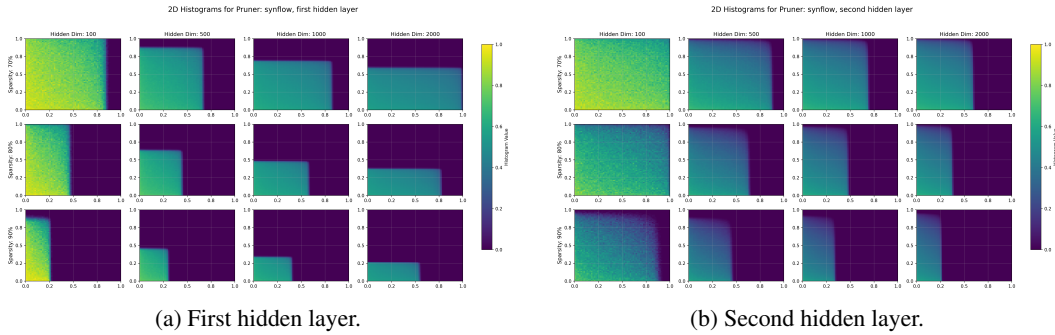


Figure 9: Graph limit of subnetworks' mask produced by Synflow pruning at different sparsity levels in 5-layer networks.

1148 D Details on numerical experiments

1149 **Experimental setup** We evaluate the relationship between Graphon NTK spectral properties and
 1150 sparse networks training dynamics using three pruning methods: Random Pruning, SNIP [42], and
 1151 SynFlow [60] at sparsity levels from 50% to 95%. Subnetworks are pruned from 4 hidden layers
 1152 network with width $n = 1024$, then trained on MNIST. Subnetworks are then trained on MNIST with
 1153 Adam Optimizer with 0.001 learning rate. All experiments are run 3 times. We illustrate the training
 1154 loss in the first 200 update steps during training.

1155 We approximate the Graphon NTK by graphon functions provided in Section 4. In particular, given
 1156 the sparsity, we generate the mask for a 4 hidden layers network with width $n = 1024$ based on
 1157 graphon functions. Then we compute the Graphon NTK based on a batch of 128 data samples from
 1158 10 classes in MNIST. Since the eigenvalues of the NTK (on the training data) quantify how much the
 1159 kernel emphasizes certain directions in function space, we analyze four spectral metrics based on
 1160 eigenvalues:

- 1161 • Eigenvalue decay rate (α): The decay exponent α describes how fast these eigenvalues fall
 1162 off, and it is approximated by fitting into the pow-law curve $\lambda_k \propto k^{-\alpha}$.
- 1163 • Effective rank $\text{trace}(\Theta)/\lambda_1$: Quantifies the functional "dimensionality" of the kernel, showing
 1164 how many independent directions meaningfully contribute to learning; lower values
 1165 indicate stronger concentration on a few key patterns
- 1166 • Spectral gap (λ_1/λ_2): The ratio between the first and second eigenvalues, revealing how
 1167 dominant the primary learning direction is compared to others; larger gaps indicate the
 1168 network will prioritize one function class extensively.
- 1169 • Energy concentration $\frac{\sum_{i=1}^k \lambda_i}{\sum_{j=1}^n \lambda_j}$: Measures what fraction of the kernel's total power resides in
 1170 the top eigenvalues; higher concentration means the kernel heavily emphasizes a few key
 1171 patterns. We use $k = 5$ in our experiments.

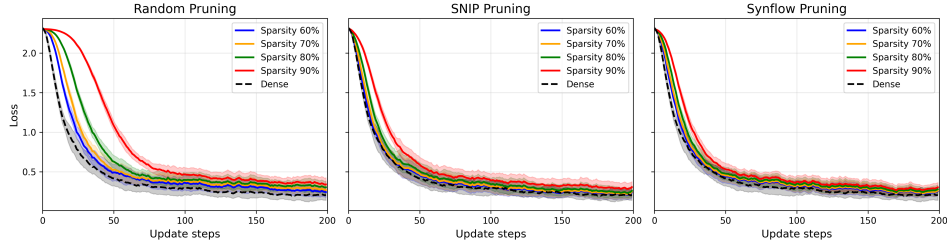


Figure 10: The training loss in the first 200 gradient update steps of training sparse networks produced by Random, SNIP, and Synflow with different sparsity levels compared with dense networks.

1172 The training curves in Figure 10 reveal distinctive learning dynamics across pruning methods that
 1173 can be elegantly explained through the Graphon NTK framework. For all methods, increased
 1174 sparsity generally slows convergence, with dense networks typically converging fastest, but the
 1175 magnitude of this effect varies significantly between approaches. Random pruning exhibits the
 1176 strongest degradation with increased sparsity, showing a clear separation between different sparsity
 1177 levels. Meanwhile, SNIP and Synflow maintain better convergence than random pruning at equivalent
 1178 sparsity levels. These differences emerge from how each pruning method shapes the graphon structure
 1179 ($\mathcal{W}^{(l)}$ functions) in the kernel formula.

1180 E Limitations and future research directions

1181 The Graphon Neural Tangent Kernel framework offers valuable insights into network pruning
1182 dynamics, but several limitations and promising research directions deserve attention.

1183 **Limitations** The infinite-width assumption underlying Graphon NTK theory creates a gap between
1184 theoretical predictions and finite networks used in practice. Real-world neural networks have limited
1185 width, and the approximation error becomes more significant at extreme sparsity levels. In our
1186 analysis, due to the computational resource constraint, we only approximate the Graphon NTK with
1187 maximum width $n = 1024$. Besides, we currently use the approximated graphons of SNIP and
1188 Synflow pruning to do the analysis since finding the exact graphon functions for these methods is
1189 non-trivial and deserves a separated work.

1190 While our analysis empirically demonstrates a correlation between graphon structure and convergence
1191 properties, the causal mechanisms deserve deeper investigation. Different pruning methods may
1192 affect various network properties simultaneously, making it difficult to isolate the specific contribution
1193 of graphon structure to observed training dynamics.

1194 **Future research directions** Several promising research directions could address these limitations
1195 and extend the graphon NTK framework:

1196 The framework could be extended to heterogeneous architectures by developing specialized graphon
1197 formulations for different architectures, e.g., transformers, convolution networks, and graph neural
1198 networks. This would significantly broaden its applicability across modern deep learning.

1199 Graphon-guided pruning algorithms could be developed that explicitly preserve favourable kernel
1200 properties rather than relying on heuristics. This would involve identifying specific graphon structures
1201 that maintain optimal gradient flow and then designing pruning methods that target these structures.
1202 Such algorithms could adaptively adjust connectivity patterns based on theoretical spectral properties
1203 of the resulting Graphon NTK.

1204 Continuous graphon optimization presents a paradigm shift from discrete mask-based approaches.
1205 Instead of pruning connections in a binary fashion, research could develop differentiable parame-
1206 terization of graphons that enable direct optimization in a continuous space. This would transform
1207 pruning from a discrete selection problem to a continuous optimization problem, potentially yielding
1208 more principled sparse networks.

1209 Graphon-informed sparse training methodologies could leverage the theoretical insights to design
1210 sparse training algorithms. By monitoring kernel properties as training progresses, these approaches
1211 could adaptively grow or prune connections, or adjust learning schedules, or even parameters to
1212 maintain optimal learning dynamics throughout the training process.