Journal of Machine Learning Research 23 (2022) 1-46 Submitted 12/20; Revised 9/21; Published 2/22

Universal Approximation in Dropout Neural Networks

Oxana A. Manita O.ZAAL.MANITA@TUE.NL
Mark A. Peletier M.A.PELETIER@TUE.NL
Jacobus W. Portegies J.W.PORTEGIES@TUE.NL
Jaron Sanders JARON.SANDERS@TUE.NL
Albert Senen–Cerda A.SENEN.CERDA@TUE.NL
Department of Mathematics & Computer Science
Eindhoven University of Technology
Eindhoven, The Netherlands

Editor: Ruslan Salakhutdinov

Abstract

We prove two universal approximation theorems for a range of dropout neural networks. These are feed-forward neural networks in which each edge is given a random $\{0, 1\}$-valued filter, that have two modes of operation: in the first each edge output is multiplied by its random filter, resulting in a random output, while in the second each edge output is multiplied by the expectation of its filter, leading to a deterministic output. It is common to use the random mode during training and the deterministic mode during testing and prediction.

Both theorems are of the following form: Given a function to approximate and a threshold $\varepsilon > 0$, there exists a dropout network that is ε-close in probability and in L^q. The first theorem applies to dropout networks in the random mode. It assumes little on the activation function, applies to a wide class of networks, and can even be applied to approximation schemes other than neural networks. The core is an algebraic property that shows that deterministic networks can be exactly matched in expectation by random networks. The second theorem makes stronger assumptions and gives a stronger result. Given a function to approximate, it provides existence of a network that approximates in both modes simultaneously. Proof components are a recursive replacement of edges by independent copies, and a special first-layer replacement that couples the resulting larger network to the input.

The functions to be approximated are assumed to be elements of general normed spaces, and the approximations are measured in the corresponding norms. The networks are constructed explicitly. Because of the different methods of proof, the two results give independent insight into the approximation properties of random dropout networks. With this, we establish that dropout neural networks broadly satisfy a universal-approximation property.

Keywords: neural networks, approximation, dropout, random neural network

1. Introduction

The class of functions that are generated by neural networks satisfies a 'universal approximation property': any given function can be approximated to arbitrary precision by such a

†. All authors contributed equally to this work.

neural network (Cybenko, 1989; Leshno et al., 1993). This property partially explains why neural networks are so effective as approximators of implicitly given functions.

It is commonly observed that the training of such networks improves upon introducing *dropout*, the random 'dropping' of edges (Goodfellow et al., 2016, Sec. 7.12). Dropout converts a deterministic network into a random one. In this paper we address the question that this observation implicitly raises: Does this randomness interfere with the universal approximation property? Or, to formulate it in the affirmative: does the class of dropout neural networks still satisfy universal approximation?

To provide a first quantification of this question, let us explain the expectation–variance split, which in the context of dropout goes back to the theoretical analysis by Wager et al. (2013). We will think of a dropout neural network as a function $\Psi : \mathbb{R}^d \times \mathbb{R}^n \to \mathbb{R}$ together with a $\{0,1\}^n$-valued random variable f. We call the components of f filter variables. We think of $\mathbb{R}^d$ as data space and $\mathbb{R}^n$ as parameter space (the space of weights and biases for the neural network). The parameters get multiplied elementwise with the vector of filter variables f. That means that when $\zeta : \mathbb{R}^d \to \mathbb{R}$ is a function we want to approximate, we try to approximate it with the stochastic function that maps x to $\Psi(x, w \odot f)$. For fixed x, the expectation–variance split reads

$$\mathbb{E}\left[\big(\Psi(x, w \odot f) - \zeta(x)\big)^2\right] = \big(\mathbb{E}[\Psi(x, w \odot f)] - \zeta(x)\big)^2 + \mathbb{V}[\Psi(x, w \odot f)]. \tag{1}$$

Here $\odot$ denotes element-wise multiplication, and throughout the paper $\mathbb{P}$, $\mathbb{E}$, $\mathbb{V}$ stand for probability, expectation, variance with respect to the filter variables f, respectively. As both terms on the right-hand side are nonnegative, both terms have to be small in order to have a good approximation. Is this possible?

Foong et al. (2020) showed that deep Rectified Linear Unit (ReLU) neural networks with node-dropout still can approximate functions arbitrary well, by showing that both the expectation term and the variance term in the expectation–variance split can be made arbitrary small (see their Theorem 3). In fact, the two terms are arbitrary small uniformly over x in the unit cube in $\mathbb{R}^d$. With this statement, Foong et al. effectively showed a universal-approximation result.

In this paper we show two universal-approximation results for wider classes of dropout neural networks. Where Foong et al. made specific use of the ReLU activation, assume Bernoulli filter variables (thus equidistributed, independent, and with finite variance), and restrict to one hidden layer, we show that the property of universal approximation holds under more general assumptions. Our distinguishing insight is that certain classes of random neural networks satisfy an algebraic property, which enables us to deal with arbitrary depth, generic activation functions, and dependent filter variables not necessarily equidistributed and possibly with infinite variance. Notably, our techniques allow for dropout of edges from the input layer. For the theorems we prove below the structural assumptions on the network reduce to the assumption that the underlying deterministic network has the universal-approximation property; necessary and sufficient conditions for the latter to hold are well-known, for example, see (Leshno et al., 1993). In addition, our main theorems allow for very general classes of filters, including the original node-based dropout (Hinton et al., 2012), the edge-based dropconnect (Wan et al., 2013), and many others, including sets of filters with strong dependence. With Theorem 1 below we show that the class of dropout networks can *exactly* match a given deterministic network, at least in expectation. With Theorem 17 below we show that we can construct networks that approximate a given function arbitrarily well, both as a random network and as a deterministic filtered network. Finally, we provide control of the error both in probability and in L^q.

1.1 Approximation by Random Neural Networks

In a deterministic context, a universal-approximation theorem for some class C is a density statement, stating that any function ζ can be approximated to arbitrary precision by neural networks in C, where the approximation is measured in some seminormed function space $(\mathcal{F}, \|\cdot\|_{\mathcal{F}})$. Such approximation statements can be generalized to a stochastic context in multiple ways. We will focus on two of these, approximation in probability and in L^q for $q \in [1, \infty)$.

Universal approximation in probability is the property that for every function $\zeta \in \mathcal{F}$ to approximate, and every $\epsilon > 0$, there exists a neural network Ψ, a weight vector w and a random vector f (all with certain extra properties to make the statement nontrivial), such that

$$\mathbb{P}\left[\|\zeta(\cdot) - \Psi(\cdot, w \odot f)\|_{\mathcal{F}} > \epsilon\right] < \epsilon.$$

A stronger statement involves approximation in L^q for $q \in [1, \infty)$: there exist Ψ, w and f such that

$$\mathbb{E}\left[\|\zeta(\cdot) - \Psi(\cdot, w \odot f)\|_{\mathcal{F}}^q\right]^{\frac{1}{q}} < \epsilon.$$

In this article we indeed show such universal approximation statements for certain classes of deep dropout neural networks.

We prove two main classes of approximation results, corresponding to the two main ways that dropout networks are used in practice. In the first class of results the network is a random object as described above, and the same network is used during training and prediction; we call this *random-approximation* dropout.[1] In the second class of results the network is trained with random networks of the form $\Psi(\cdot, w \odot f)$, but for prediction the deterministic network $\Psi(\cdot, w \odot \mathbb{E}f)$ is used in which the filters are replaced by their expectations. We call this type of dropout *expectation-replacement.*

1.2 Main Results 1: Random-approximation

We start with uniform *random-approximation*, that is the property that any function ζ in an appropriate set can be approximated by random networks of the form $\Psi(\cdot, w \odot f)$. At the highest level the proof strategy is the same as in Foong et al. and consists of the following three steps. Given a function $\zeta \in \mathcal{F}$ to be approximated:

1. Approximate ζ by a neural network $\Psi(\cdot, w)$ using classical deterministic universal approximation results (e.g., Leshno et al. (1993));

2. Use $\Psi(\cdot, w)$ to construct a larger, random dropout neural network $\tilde{\Psi}(\cdot, \tilde{w} \odot \tilde{f})$ that matches $\Psi(\cdot, w)$ in expectation;

1. This has also been called *Monte Carlo dropout* because of the close connection with Monte Carlo estimation of integrals (Gallicchio and Scardapane, 2020).

3. Construct an even larger random neural network $\widehat{\Psi}(\cdot, \widehat{w} \odot \widehat{f})$ consisting of many independent copies of the network $\tilde{\Psi}(\cdot, \tilde{w} \odot \tilde{f})$ to obtain an approximation of ζ that is close in expectation and also has small variance.

We consider Step 1 as given by existing results, and Step 3 is a standard procedure. The novelty of this paper for *random-approximation* lies in Step 2, which we describe in the rest of this section.

Step 2 is based on an algebraic property of common classes of neural networks, which is illustrated by the following simpler version of the central theorem. We write 2^n for the collection of subsets of $\overline{1,n} = \{1, \dots, n\}$, and for any such a subset U, we write $\mathbf{1}_U \in \{0,1\}^n$ for the vector with entries $(1_{j\in U})_{j\in\overline{1,n}}$.

Theorem 1 *Let $\Psi : \mathbb{R}^d \times \mathbb{R}^n \to \mathbb{R}$ be any given function. Let $(f^U)_{U\in 2^n}$ be a collection of $\{0,1\}^n$-valued random variables indexed by subsets $U \in 2^n$ such that for every U*

$$\mathbb{P}[f^U = (1, \dots, 1)] > 0.$$

Then there exist constants $(a_U)_{U\in 2^n}$, independent of w, such that for all w,

$$\mathbb{E}\left[\sum_{U\in 2^n} a_U \Psi\big(\cdot, (w \odot \mathbf{1}_U) \odot f^U\big)\right] = \Psi(\cdot, w). \tag{2}$$

This theorem should be read as follows. The right-hand side in (2) plays the role of a deterministic function that we want to approximate. The left-hand side is the expectation of a linear combination of many copies of $\Psi(\cdot, w)$. Each copy has two 'dropout' modifications: the vector $\mathbf{1}_U$ implements a deterministic dropout, and the random filter variables f^U a stochastic one. With a view to generality, the random filter vector f^U is allowed to be a different random vector for each subset U of edges, but note that the distribution of f^U on $\{0,1\}^n$ can be completely unrelated to the subset $U \subset \overline{1,n}$; the subset U only serves as label.

The theorem establishes the following fact: for *any* collection of random filter variables f^U, for *any* function Ψ, for *any* parameter point w, the function $\Psi(\cdot, w)$ can be matched exactly by the expectation of a sum of filtered versions of the same function. The important caveat is that one needs to take into account all reduced versions of the functions Ψ, i.e., the whole hierarchy of deterministically modified versions indexed by subsets U.

Theorem 1 suggests a special role for 'classes of networks', with the property that given a 'network' $\Psi(\cdot, w)$ we can in some sense define a new (random) network $\tilde{\Psi}(\cdot, \tilde{w} \odot \tilde{f})$ by

$$\tilde{\Psi}(\cdot, \tilde{w} \odot \tilde{f}) := \sum_{U\in 2^n} a_U \Psi(\cdot, w \odot \mathbf{1}_U \odot f^U). \tag{3}$$

To formalize this, we assume that we have chosen a set DDNN (a 'set of random neural networks'), which can be any collection of tuples (n, Ψ, f) that satisfy the following properties:

(i) $n \in \mathbb{N}$ is a natural number;

(ii) $\Psi : \mathbb{R}^d \times \mathbb{R}^n \to \mathbb{R}$ is a function such that for every $w \in \mathbb{R}^n$, $\Psi(\cdot, w) \in \mathcal{F}$;

(iii) f is a $\{0,1\}^n$-valued random variable such that

$$\mathbb{P}[f = (1, \dots, 1)] > 0. \tag{4}$$

Moreover, we assume that DDNN is closed under linear, independent combinations. By this we mean that whenever $a, b \in \mathbb{R}$ and (m, Φ, f) and (n, Ψ, g) are in DDNN, then also $(n', a\Phi + b\Psi, h) \in \mathsf{DDNN}$ with $n' \geq n + m$ where h is an $\{0,1\}^{n'}$-valued random variable that is the independent concatenation of f and g, and $a\Phi + b\Psi : \mathbb{R}^d \times \mathbb{R}^{n'} \to \mathbb{R}$ is given by

$$(x, (w_1, w_2)) \mapsto a\Phi(x, w_1) + b\Psi(x, w_2).$$

This closure assumption implies that a definition of the form (3) is meaningful.

The range of possible classes DDNN satisfying these requirements is vast. Typical examples are neural networks with node-dropout, as originally introduced by Hinton et al. (2012), and *dropconnect*, as introduced by Wan et al. (2013), but many other choices also are possible. Note that the function Ψ may be extremely general, implying that there are no restrictions on e.g., the form of the activation function or the structure of the network. In fact, nothing in the requirements on DDNN restricts to functions Ψ generated by neural networks; other approximation methodologies may also be used, for instance based on Fourier or wavelet expansions. See Section 2 for a detailed description of DDNN.

By combining Theorem 1 with the law of large numbers we then find Corollary 2 below, which expresses the following insight: if the class DDNN is rich enough to approximate any function in $\mathcal{F}$ when all filter variables are set to 1, then any function in $\mathcal{F}$ can also be approximated by a (random) dropout neural network in DDNN.

Corollary 2 *Let $\zeta \in \mathcal{F}$ and $\epsilon > 0$. Assume there exists a $(m, \Phi, g) \in \mathsf{DDNN}$ and a $v \in \mathbb{R}^m$ such that $\|\Phi(\cdot, v) - \zeta\|_{\mathcal{F}} < \epsilon/2$. Then there exists a $(n, \Psi, f) \in \mathsf{DDNN}$ and a $w \in \mathbb{R}^n$ such that*

$$\mathbb{P}[\|\Psi(\cdot, w \odot f) - \zeta\|_{\mathcal{F}} > \epsilon] < \epsilon \tag{5}$$

and

$$\mathbb{E}\left[\|\Psi(\cdot, w \odot f) - \zeta\|_{\mathcal{F}}^q\right]^{\frac{1}{q}} < \epsilon.$$

Section 4 is devoted to these results, but develops them in more generality. There we also give some examples and calculate the coefficients a_U explicitly for the case of independent Bernoulli filters.

1.3 Main Results 2: Expectation-replacement

In the previous section we considered dropout neural network to be a random object, both during training and during prediction. By contrast, it is common practice to choose the filter variables to be *random* during training and to be *deterministic* during prediction and equal to their expectations; see e.g., Goodfellow et al. (2016, Sec. 7.12). We call this *expectation-replacement* dropout, and Corollary 2 above does not say anything about this situation. In fact, we show in Example 5 that the construction at the heart of Corollary 2 may lead to networks that are 'bad approximators' in this specific sense: given a function ζ, the constructed networks approximate ζ with high probability with random filters, but do not approximate ζ at all when replacing the filters by their expectations.

To address this, we describe in Section 4 the construction of dropout neural networks that approximate not only in probability and in L^q, but also in this expectation-replacement sense. As in the case of Corollary 2, the construction builds on existing density results for deterministic networks: we start with a given deterministic neural network $\Psi(\cdot, w)$ that is close to a given function ζ. Differently from Corollary 2, however, the nonlinearity of Ψ forces us to apply the law of large numbers to each edge (or weight in this context) separately, instead of simultaneously for the whole network.

In the construction in Section 4 we therefore iteratively replace each edge in the deterministic neural network Ψ by a set of parallel edges, with edge-weights w taken from the original edge, and with independent filter variables on each of them. In this way we can use the law of large numbers to obtain convergence estimates for each edge separately, and then combine these estimates into a single convergence estimate for the whole network.

The convergence estimate for a single edge arises from the following statement (which is a simplified version of Lemma 14). It describes how the error encountered by averaging N independently filtered edges can be controlled in probability. At the same time it also allows for small perturbations of the inputs to this edge. This latter perturbation freedom is needed in order to apply this lemma progressively, moving from edge to edge through the network.

Lemma 3 *Consider any continuous function $\sigma : \mathbb{R}^m \to \mathbb{R}^m$ and let $W \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$. Let $\{F^i\}_{i\in\overline{1,N}}$ be a collection of independent copies of a random matrix $F \in \{0,1\}^{m\times n}$. Then for every $K > 0$ there exists a $\delta > 0$ such that*

$$\sup_{x\in\overline{B(0,K)}} \sup_{(\tilde{x}^i)\in B(x,\delta)^N} \left| \sigma\left(\frac{1}{N} \sum_{i=1}^{N} (W \odot F^i)\tilde{x}^i + b \right) - \sigma\Big((W \odot \mathbb{E}F^i)x + b \Big) \right| \tag{6}$$

converges to zero in probability as $N \to \infty$.

In Section 4 this construction is described in detail. A separate part of this description is how to connect the resulting dropout neural network to the inputs of the original layer; for this we introduce a single additional layer that implements this connection.

The main Theorem 17 allows for a wide range of choices of activation functions and filter-variable distributions. For example, the following are simple, concrete corollaries for a ReLU activation function with dropconnect and node-dropout respectively.

Corollary 4 *Take $\mathcal{F}$ to be the space of continuous functions $\mathbb{R}^d \to \mathbb{R}$, and endow it with a seminorm $\|\cdot\|_{\mathcal{F}}$ equal to supremum of the function on the unit cube. Then for every $\zeta \in \mathcal{F}$ and every $\epsilon > 0$ there exists a dropconnect ReLU neural network (Ψ, f) and a parameter vector w such that*

$$\mathbb{P}\left[\left\| \Psi(\cdot, w \odot f) - \zeta \right\|_{\mathcal{F}} > \epsilon \right] < \epsilon \tag{7}$$

and

$$\mathbb{E}\left[\|\Psi(\cdot, w \odot f) - \zeta\|_{\mathcal{F}}^q \right]^{\frac{1}{q}} < \epsilon,$$

while

$$\|\Psi(\cdot, w \odot \mathbb{E}[f]) - \zeta\|_{\mathcal{F}} < \epsilon.$$

Corollary 5 Corollary 4 *also holds when using node-dropout instead of dropconnect.*

Note that where the construction of the previous section applied to a very wide class of functions Ψ—not only those generated by neural networks—the construction underlying Corollaries 4 and 5 depends in a detailed manner on the fact that Ψ has the structure of a neural network.

Finally, we want to emphasize that constructing a network that approximates well when filters are replaced by their expectations (the last bound in Corollaries 4 and 5) is not difficult due to the universal approximation property for deterministic networks: it is enough to scale the coefficients. However the obtained network may have no relation to the training process. In order to overcome that, we need *simultaneous* approximation as a random network and as a deterministic network after replacing filters by their expectations.

1.4 Random-approximation *vs.* Expectation-replacement Dropout

Using a *random* neural network to approximate a given *deterministic* function is non-trivial; the variance of the network needs to be reduced while matching the expectation, as described in Section 1.2.

In expectation-replacement dropout, however, the networks used in prediction are deterministic, and this difficulty is absent. In fact, the difference between training and prediction is the reason we include expectation-replacement in this paper.

This difference between training and prediction poses an intriguing question. Suppose that the dropout training algorithm yields a parameter point w^*. During this training, the algorithm has observed random networks $\Psi(\cdot, w \odot f)$, but it has never observed the deterministic network $\Psi(\cdot, w \odot \mathbb{E}f)$. Why, then, should the result w^* of the dropout training then generate a good deterministic network $\Psi(\cdot, w^* \odot \mathbb{E}f)$? Example 5 confirms that this method may fail badly.

At the same time, expectation-replacement dropout is both very widespread and very successful; see e.g., Goodfellow et al. (2016, Sec. 7.12) or Labach et al. (2019). How can these two observations be reconciled?

The results of Section 4 and e.g., Corollary 4 or 5 provide a partial answer to this question. We show that dropout neural networks have sufficient representational capacity to approximate well simultaneously in probability, in L^q, *and* in the expectation-replacement sense. While this does not explain why any given training algorithm finds parameter points that approximate well in the expectation-replacement sense, at least it shows that the contrast between random training and deterministic prediction is not an obstacle to good performance.

1.5 Related Literature

The universal approximation property for neural networks is one of the fundamental properties and essentially determines whether the whole training process of the network makes sense: if the algorithmically generated functions don't form a dense set in the function space of interest, the approximation problem is ill-posed. Therefore establishing the universal approximation property for different classes of networks has been an active research area in the last decades. However, most classes of networks for which there is a universal approximation property established do not include, for example, node-dropout or dropconnect neural network.s

The first universal approximation theorem for neural networks with a sigmoidal activation function can be found in Cybenko (1989)'s paper, and this canonical work led to much follow-up research. Several years later Hornik (1991) showed that the universal approximation property relies more on a neural network's architecture than on the specific use of sigmoid activation functions. Moreover, Leshno et al. (1993) established that deep, feed-forward neural networks require a nonpolynomial activation function in order for a universal approximation theorem to hold. Makovoz (1996, 1998) used the so-called probabilistic method to prove the existence of a deterministic function that suitably approximates a target function in deterministic neural networks.

Approximately at the same time the study of random networks started. White (1989)'s paper on "QuickNet" is one of the first works where universal approximation is mentioned (but not proved) side by side with a neural network algorithm in which random hidden nodes are placed.

The class of networks with random weights and biases, called Random Vector Functional-Link Nets, was introduced in 1994 by Pao et al. (1994). Igelnik and Pao (1995) proved a universal approximation property of these networks, by showing that the span of the node functions is almost surely asymptotically dense in the many-node limit. This result does not apply to dropout schemes since in the dropout setup the randomness is applied after choosing coefficients.

Gelenbe et al. (1999a,b) introduced a class of neural networks that relies on a fixed neural network topology on top of which neurons forward positive and negative signals (spikes) at random points in time based on their own "potential". Specifically, they gave a constructive proof of the universal approximation theorem for such stochastic neural networks networks in steady state. This class of networks also doesn't cover the node-dropout or dropconnect cases due to the different dynamics assumed; moreover a dropout neural network is trained randomly, but typically operated deterministically.

Rahimi and Recht (2008) investigated uniform approximation of functions with random bases. This is a particular case of a so-called random feature method, in which the parameters are split in two groups: parameters in one group are taken randomly (and not tuned), and the other part is trained to achieve best approximation. Therefore these results also don't cover node-dropout or dropconnect since for the latter algorithms all parameters are trained.

Another commonly used class of neural networks is the *mixture of experts* model. The idea is that for different input regions different, typically simpler, networks (learners) are used for prediction. The choice is performed by the *gating network*; training of the model consists then of training individual learners together with training the gating network. Nguyen et al. (2016) proved a universal approximation theorem for a mixture-of-experts model, and Nguyen (2017) subsequently generalized their findings to allow for so-called Gaussian gating.

De Bie et al. (2019) considered a network architecture that can handle probability measures as input and output. They proved the universal approximation in Wasserstein metric for continuous maps from the space of measures into itself. Our results are more specific, and not covered by this result, since we study a different (more restricted) approximation scheme.

As mentioned in the introduction, Foong et al. (2020) show a universal approximation property for random-approximation dropout networks (see their Theorem 3). We recover this result as Corollary 2 when identifying $h \equiv 0$. Another difference is that we allow for activation functions other than ReLU activation functions, and consider a stronger sense of approximation.

Finally, we refer interested readers to the following surveys to fully complete their picture of known results. A survey of approximation-theoretic problems was written by Pinkus (1999); a recent survey by Elbrächter et al. (2020) contains a comparison of approximation properties for finite-width and finite-depth networks. Several uniform approximation results for random neural networks can be found in Timotheou (2010)'s Section 5.4. Approximation literature for random neural networks was also summarized by Yin (2019).

1.6 Structure of this paper

Definitions of dropout neural networks are given in Section 2. In Section 3 we show universal approximation results for random-approximation dropout, whereas Section 4 is devoted to universal approximation results for expectation-replacement dropout. We discuss our results in Section 5 and conclude in Section 6.

2. Specification of Dropout Neural Networks

In the introduction, we considered general functions $\Psi : \mathbb{R}^d \times \mathbb{R}^n \to \mathbb{R}$ together with a $\{0,1\}^n$-valued random variable f (Section 1.2), and more specific functions Ψ that arise from a neural network (Section 1.3). In this section, we specify this neural network structure and introduce the corresponding notation.

2.1 Neural Networks

We specify a (feedforward) neural network as a special type of parametrized function $\Psi : \mathbb{R}^d \times \mathbb{R}^n \to \mathbb{R}$ from an input vector space $\mathbb{R}^d$ to $\mathbb{R}$, parametrized by vectors in $\mathbb{R}^n$. The function is special in that it is assumed to be the composition of multiple functions of much simpler type

$$\Psi(\cdot, w) = \Psi_L\left(\cdot, w^{(L)}\right) \circ \Psi_{L-1}\left(\cdot, w^{(L-1)}\right) \circ \cdots \circ \Psi_1\left(\cdot, w^{(1)}\right). \tag{8}$$

Here, L is an integer, the parameter w is the concatenation of the individual parameter vectors $w^{(j)} = (W^{(j)}, b^{(j)})$ for $j = \overline{1, L}$, which in turn consist of a $d_j \times d_{j-1}$ *weight matrix* $W^{(j)}$ and a bias vector $b^{(j)} \in \mathbb{R}^{d_j}$. We set $d_0 = d$ and $d_L = 1$.

In (8) every Ψ_j is a function from $\mathbb{R}^{d_{j-1}}$ to $\mathbb{R}^{d_j}$ given by

$$\Psi_j(x, w^{(j)}) := \sigma_j\left(W^{(j)}x + b^{(j)}\right), \tag{9}$$

where the function $\sigma_j : \mathbb{R} \to \mathbb{R}$ is called the *activation function.* The activation function is applied elementwise.

2.2 Dropout Neural Networks

A dropout neural network consists of a neural network $\Psi : \mathbb{R}^d \times \mathbb{R}^n \to \mathbb{R}$ as above together with a random vector $f \in \{0,1\}^n$. The components of f are called filter variables. The network Ψ, the filter variables f, and a parameter vector $w \in \mathbb{R}^n$ together form a stochastic function from $\mathbb{R}^d$ to $\mathbb{R}$ given by

$$x \mapsto \Psi(x, w \odot f).$$

For the constructions later in the article, we recall what we precisely mean by random variables. Throughout the article, $(\Omega, \mathcal{F}_\Omega, \mathbb{P})$ is an arbitrary, rich enough, probability space. Whenever we write *random variable*, *random vector* or *random matrix*, we mean a measurable function defined on this probability space.

2.2.1 Node-dropout

In the original version of dropout filter variables acted on *nodes* of the network (Hinton et al., 2012). In this paper the filter variables act on edges instead. The original version, which we call *node-dropout*, can be represented in the edge-based version of this paper as follows.

The filter variables are partitioned into various blocks: filter variables are in the same block if and only if they multiply an element in the same column in the same weight matrix, or they multiply elements of the same bias vector. Filter variables in the same block always attain the same value, i.e., with probability one. Filter variables in different blocks are independent. We will use the convention that filter variables that multiply biases are always on, whereas filter variables that multiply elements of weight matrices are on, i.e., equal to 1, with probability $1-p$ for some $0 \le p < 1$.

We can understand node-dropout from the previous description in the notation of (9). For any $j = 1, \dots, L$, choose probabilities p^j, and let $f_1^j, \dots, f_{d_j}^j$ be independent Bernoulli filters with probability $1-p^j$. Let $D_j \in \mathbb{R}^{d_j \times d_j}$ be the diagonal matrix with entries $f_1^j, \dots, f_{d_j}^j$ in the diagonal. If we then arranging all nodes per block, then node-dropout implements for $j = 1, \dots, L$,

$$\Psi_j(\cdot, w^{(j)} \odot f^{(j)}) = \sigma_j \left(W^{(j)} D^j (\cdot) + b^{(j)} \right). \tag{10}$$

Note that if $p^1 > 0$ then with positive probability an input is masked. For this reason we call the case $p^1 > 0$ node-dropout *with* dropout on the inputs. We call the case $p^1 = 0$ node-dropout *without* dropout on the inputs.

2.2.2 Dropconnect

Dropconnect is another dropout regularization scheme (Wan et al., 2013). Although Wan et al. (2013) also allowed for dropout of biases, we will use the term *dropconnect* for the dropout neural network in which only the matrices $W^{(j)}$ are filtered. This is achieved by choosing the filter variables multiplying the biases to be equal to 1 with probability one.

We can understand dropconnect in the notation of (9). For $j = 1, \dots, L$, let $F^{(j)} \in \mathbb{R}^{d_{j+1} \times d_j}$ be random matrices composed of entries $(F^j)_{ik}$, all of which are mutually independent Bernoulli random variables with the same success probability $1-p$. Dropconnect then implements for $j = 1, \dots, L$,

$$\Psi_j(\cdot, w^{(j)} \odot f^{(j)}) = \sigma_j \left((W^{(j)} \odot F^{(j)})(\cdot) + b^{(j)} \right). \tag{11}$$

3. Universal Approximation for Random Approximation Dropout

The aim of this section is to derive the abstract universal approximation statement for random-approximation dropout already mentioned in the introduction (Corollary 2).

3.1 Key Approximation Result

The following theorem is Theorem 1 in the introduction, extended with a convergence statement.

Theorem 6 *Let $(\mathcal{F}, \|\cdot\|_{\mathcal{F}})$ be a seminormed vector space of functions from $\mathbb{R}^d$ to $\mathbb{R}$. Let $\Psi : \mathbb{R}^d \times \mathbb{R}^n \to \mathbb{R}$ be a given function such that $\Psi(\cdot, w) \in \mathcal{F}$ for every $w \in \mathbb{R}^n$. Let $(f^U)_{U \in 2^n}$ be a collection of $\{0,1\}^n$-valued random variables indexed by subsets $U \in 2^n$, such that for every U*

$$\mathbb{P}[f^U = (1, \dots, 1)] > 0. \tag{12}$$

Then there exist constants $(a_U)_{U \in 2^n}$ independent of w such that

$$\mathbb{E}\left[\sum_{U \in 2^n} a_U \Psi(\cdot, (w \odot \mathbf{1}_U) \odot f^U) \right] = \Psi(\cdot, w). \tag{13}$$

In particular, by the weak law of large numbers, if $f^{i,U}$ are independent copies of f^U, then as $M \to \infty$,

$$\frac{1}{M} \sum_{i=1}^{M} \sum_{U \in 2^n} a_U \Psi(\cdot, (w \odot \mathbf{1}_U) \odot f^{i,U}) \to \Psi(\cdot, w) \tag{14}$$

in probability in $(\mathcal{F}, \|\cdot\|_{\mathcal{F}})$ and in L^q for every $q \in [1, \infty)$.

A proof of Theorem 6 can be found in Appendix A.1. The main observation in Theorem 6 is the existence of the constants $(a_U)_{U \in 2^n}$. This purely algebraic statement follows by induction, as explained by Lemma 19 in Appendix A.1. From Theorem 6, it follows that one can see a dropout neural network as a linear combination of dropout networks with weights $(w \odot \mathbf{1}_U)_{U \in 2^n}$, such that the linear combination equals the original neural network in expectation as shown in (13).

To get a dropout neural network that is close to the original network in probability, in (14) one makes a large average of independent copies of the dropout network that approximates the original network in expectation. The convergence in probability of (14) follows then from the weak law of large numbers. The convergence in L^q finally follows because the expectation is uniformly bounded in $\mathcal{F}$ for any realization of the filter variables $f^{i,U}$, so that the convergence in probability immediately implies the convergence in L^q by dominated convergence.

Note that the number of parameters in this construction increases exponentially, which limits the potential applicability of this theorem. On the other hand, for particular cases the coefficients can be calculated explicitly, as it will be shown in Section 3.4.

3.2 Examples

We further illustrate the construction of Theorem 6 with the following examples:

Example 1 (One-hidden-layer dropconnect networks) *Consider the function* $\Psi : \mathbb{R}^d \times \mathbb{R}^n \to \mathbb{R}$ *given by*

$$\Psi(x, w) := \sum_{j=1}^{N} c^j \sigma(w^j x + b^j) \tag{15}$$

where the activation function $\sigma : \mathbb{R} \to \mathbb{R}$ *is continuous with* $\sigma(x) \to 0$ *as* $x \to -\infty$ *and* $\sigma(x) \to 1$ *as* $x \to \infty$*. In* (15) *we have biases* $b_j \in \mathbb{R}$ *and weights made up from the constants* $c^j \in \mathbb{R}$ *and the* $1 \times d$*-matrices* w^j*.*

The well-known result by Cybenko (1989) implies that the class of all such functions is dense in $C([0,1]^d)$ *endowed with the supremum norm. An example of an approximation by functions in* (15) *is depicted in Figure 1.*

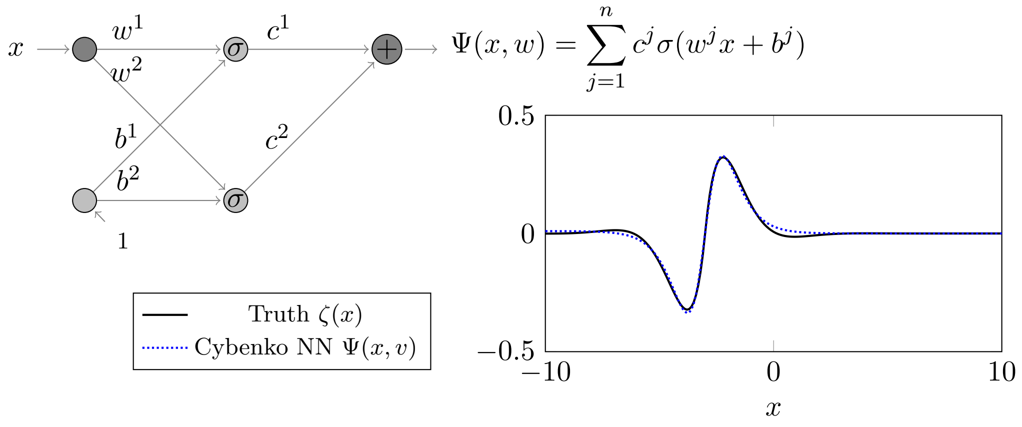

Figure 1: A Cybenko neural network as in (15), trained to approximate a function ζ; here, $n = 2, d = 1$, and $\zeta(x) = \sin(x+3)\exp|x-3|$.

We suppose that the distribution of the filters follows the case of dropconnect, as described in Section 2.2.2. Theorem 6 *directly yields that by choosing appropriate weights* $c^{j,U}$ *and weight matrices* $w^{j,U}$*, the one-hidden-layer dropconnect network given by*

$$\frac{1}{M} \sum_{i=1}^{M} \sum_{U \in 2^n} \sum_{j=1}^{N} a_U c^{j,U} g^{j,U} \sigma\left((w^{j,U} \odot f^{j,U})x + b^j\right) \tag{16}$$

can be chosen to be close to Ψ *in* L^q *for large* M*. Here* $g^{j,U}$ *are independent Bernouilli random variables, and* $f^{j,U}$ *are random vectors with independently Bernoulli-distributed components, all with success probability* $1-p$*. This result is illustrated by Figures 2 and 3, where for simplicity we have used filters only on the weights* $w^{j,U}$*, while leaving the biases* b^j *and* c^j *with constant filters* 1*. Figure 2 shows a single realization of the neural network in* (13) *with dropconnect while in Figure 3 a 'blow up'—the average of* M *independent copies of the network in* (16)*—of the previous construction is depicted.*

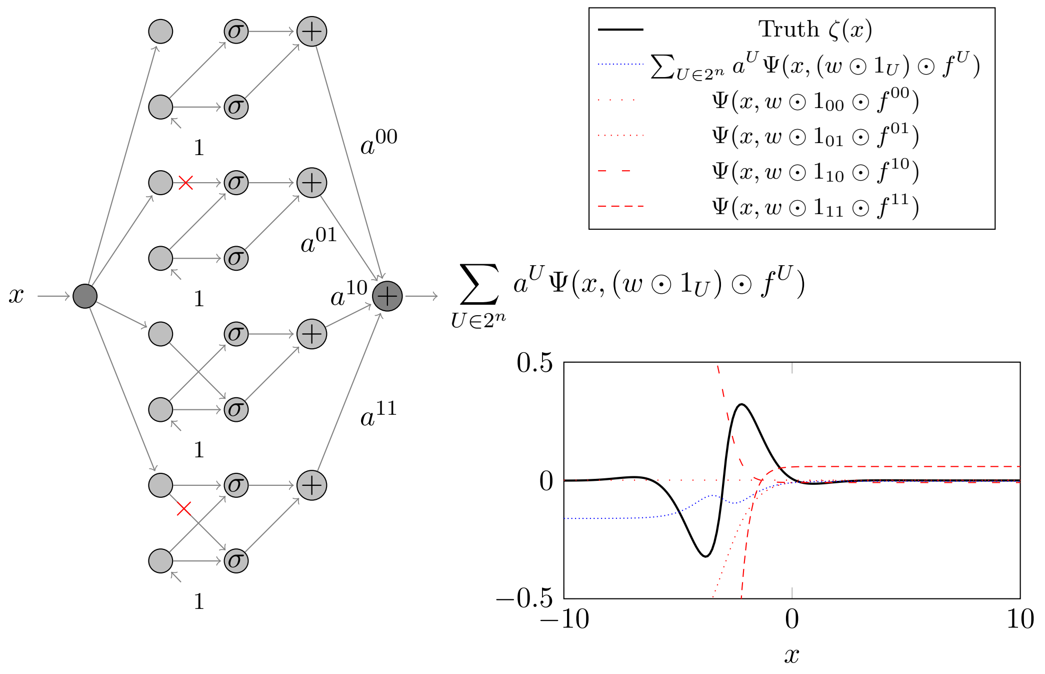
x
σ
1
a^{00}
a^{01}
a^{10}
a^{11}
$\sum_{U\in 2^n} a^U\,\Psi(x,(w\odot 1_U)\odot f^U)$
Truth $\zeta(x)$
$\sum_{U\in 2^n} a^U\,\Psi(x,(w\odot 1_U)\odot f^U)$
$\Psi(x,w\odot 1_{00}\odot f^{00})$
$\Psi(x,w\odot 1_{01}\odot f^{01})$
$\Psi(x,w\odot 1_{10}\odot f^{10})$
$\Psi(x,w\odot 1_{11}\odot f^{11})$
0.5
0
−0.5
−10
0
10
x

Figure 2: A single realization of the random neural network in (13) using Dropconnect. Based off the trained Cybenko neural network in Figure 1, for simplicity, we only apply dropout to the weights w^j of (15), which we denote by w and correspond to the edges joining the nodes connected to the input x. With $n = 2$ and $d = 1$, there are four different random neural networks with their respective independent filters. All of them use as base $\Psi(\cdot, w)$ in Figure 1. In this realization, some of the edges are filtered, which are depicted with red crosses. The explicit coefficients a_U used for dropconnect are computed in Section 3.4.

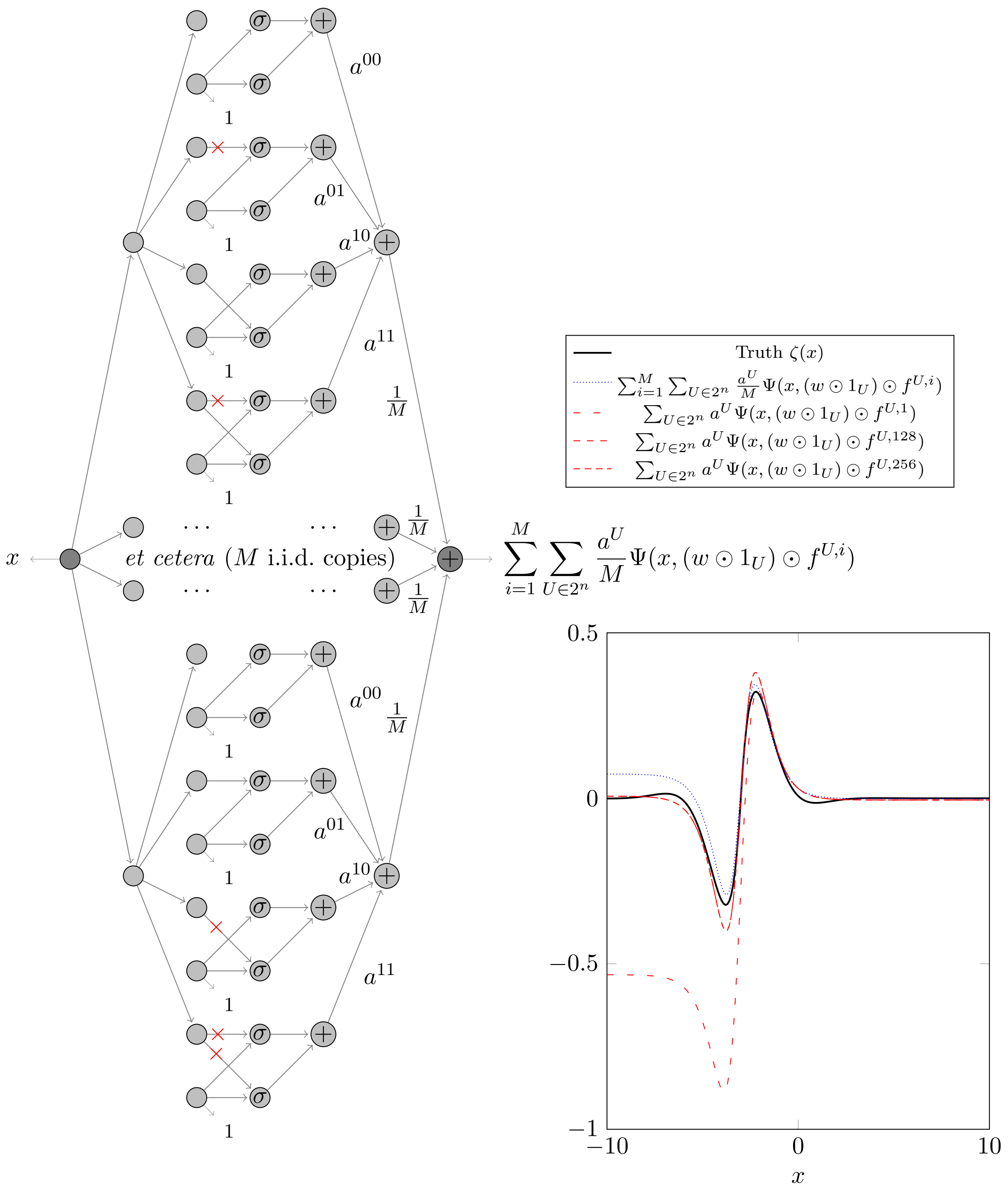
x
a^{00}
a^{01}
a^{10}
a^{11}
1
σ
$\frac{1}{M}$
et cetera (M i.i.d. copies)
$\sum_{i=1}^{M}\sum_{U\in 2^n}\frac{a^U}{M}\Psi(x,(w\odot 1_U)\odot f^{U,i})$
Truth $\zeta(x)$
$\sum_{i=1}^{M}\sum_{U\in 2^n}\frac{a^U}{M}\Psi(x,(w\odot 1_U)\odot f^{U,i})$
$\sum_{U\in 2^n}a^U\Psi(x,(w\odot 1_U)\odot f^{U,1})$
$\sum_{U\in 2^n}a^U\Psi(x,(w\odot 1_U)\odot f^{U,128})$
$\sum_{U\in 2^n}a^U\Psi(x,(w\odot 1_U)\odot f^{U,256})$
0.5
0
−0.5
−1
−10
0
10
x

Figure 3: An approximation of ζ with a large neural network using dropconnect, based off the base Cybenko neural network in (15) depicted in Figure 1. Adding many independent copies of the network from Figure 2, we are leveraging the law of large numbers as in (16). Different independent copies of the network may have a different realization of the filters, which is here depicted by the red crosses on the edges.

In a similar way, we can also consider more general dropconnect networks.

Example 2 (Dropconnect networks) *Consider a deep neural network $\Psi : \mathbb{R}^d \times \mathbb{R}^n \to \mathbb{R}$ as introduced in* (8) *with dropconnect filters as described in Section 2.2.2. Here, the filter variables, i.e., the components of $f^{i,U}$ in* (8)*, are i.i.d. Bernoulli distributed with success probability $1-p$ if they multiply elements of the weight matrices $W^{(j)}$, and are equal to* 1 *if they multiply biases $b^{(j)}$.*

Let $w \in \mathbb{R}^n$. We choose for $\mathcal{F}$ the vector space of continuous functions on $\mathbb{R}^d$, endowed with the supremum seminorm over the closed unit cube. Then the dropconnect random network in (14) *is for large M close to the network $\Psi(\cdot, w)$ in L^q.*

Example 3 (Node-dropout networks) *Consider again the deep neural network in* (8) *with node-dropout as described in Section 2.2.1. The random neural network in* (14) *is then again a node-dropout neural network. In this way, we recover Foong et al. (2020)'s Theorem 3 (with $h \equiv 0$), which for ReLU activation functions and a target function ζ bounds*

$$\sup_{x \in [0,1]^d} \operatorname{Var}(\zeta(x) - \Psi(x, w \odot f)). \tag{17}$$

When $\mathcal{F}$ is the space of continuous functions with supremum norm, (17) *can be bounded by a constant times the square of the L^2-norm. Hence,* Theorem 6 *approximates in a stronger sense, namely, in L^q for any $q \in [1, \infty)$. Moreover,* Theorem 6 *also allows for activation functions other than ReLU.*

Example 4 (Dropout networks with dropout on *input*) *In contrast, if there is also dropout on the input, then the neural network in* (14) *is* not *again a dropout neural network with dropout on the inputs. Results by Foong et al. (2020) imply that in general neural networks with dropout on the* input *cannot satisfy a universal approximation property.*

We remark that this kind of stochastic network is not *a dropout neural network as defined in Section 2.2 as the following example shows: Suppose that $\Psi_1, \Psi_2 : \mathbb{R}^d \times \mathbb{R}^n \to \mathbb{R}$ are two different dropout neural networks with weights w_1, w_2 and with respective filter random variables f, g with values in $\{0,1\}^n$. Then we can define the dropout neural network Ψ with value*

$$\Psi(x, (w_1 \odot f, w_2 \odot g)) = \Psi_1(x, w_1 \odot f) + \Psi_2(x, w_1 \odot g). \tag{18}$$

Suppose that, additionally, we add independent filters h_1 and h_2 with values in $\{0,1\}^d$ to Ψ_1, Ψ_2 for their respective inputs. Then, $\Psi_1(x \odot h_1, w_1) + \Psi_2(x \odot h_2, w_2)$ is not necessarily of the type $\Psi(x \odot h, (w_1, w_2))$ for some random variable h with values in $\{0,1\}^d$.

As the above examples illustrate, a crucial aspect of whether a certain class of dropout neural networks (such as dropconnect or node-dropout) satisfy a universal approximation property, is whether linear, independent combinations of such networks are again networks in the same class. On the other hand, many details of the neural networks, such as them being a composition of simpler functions, are irrelevant for the proof of Theorem 6.

3.3 The Classes DDNN

In the introduction we introduced classes DDNN of tuples (n, Ψ, f) that are closed under linear, independent combinations as the basic objects with which we want to approximate a given function $\zeta \in \mathcal{F}$.

The convergence statement (14) of Theorem 6 then immediately implies Corollary 7, which was already given as Corollary 2. It expresses that if the class DDNN is rich enough to approximate any function in $\mathcal{F}$ when all filter variables are set to 1 in the event in (4), then for every function in $\mathcal{F}$ there exists a dropout neural network such that with high probability with regards to the filter variables, the dropout neural network also approximates the function.

Corollary 7 *Let $\zeta \in \mathcal{F}$ and $\epsilon > 0$. Assume there exists a $(m, \Phi, f) \in \mathsf{DDNN}$ and a $v \in \mathbb{R}^m$ such that $\|\Phi(\cdot, v) - \zeta\|_{\mathcal{F}} < \epsilon/2$. Then there exists a $(n, \Psi, g) \in \mathsf{DDNN}$ and a $w \in \mathbb{R}^n$ such that*

$$\mathbb{P}[\|\Psi(\cdot, w \odot g) - \zeta\|_{\mathcal{F}} > \epsilon] < \epsilon \tag{19}$$

and

$$\mathbb{E}\left[\|\zeta(\cdot) - \Psi(\cdot, w \odot g)\|_{\mathcal{F}}^q\right]^{\frac{1}{q}} < \epsilon.$$

The proof of Corollary 7 can be found in Appendix A.2. This corollary can be then combined with deterministic universal approximation properties of certain classes of neural networks to obtain concrete universal approximation properties of dropout neural networks. For instance, because both the class of node-dropout networks and the class of dropconnect networks defined in Sections 2.2.1 and 2.2.2 form examples of a set DDNN, we obtain the following universal approximation property by combining Corollary 7 with the universal approximation result in Leshno et al. (1993)'s Proposition 1.

Corollary 8 *Assume μ is a nonnegative probability measure on $\mathbb{R}^n$ with compact support, absolutely continuous with respect to the Lebesgue measure. Take $\mathcal{F} = L^r(\mu)$ for some $r \in [1, \infty)$. Assume that the activation function $\sigma : \mathbb{R} \to \mathbb{R}$ is not equal to a polynomial almost everywhere. Then for every $\epsilon > 0$ there exists a one-hidden-layer dropconnect neural network (n, Ψ, g) such that*

$$\mathbb{P}[\|\Psi(\cdot, w \odot g) - \zeta\|_{L^r(\mu)} > \epsilon] < \epsilon, \tag{20}$$

and

$$\mathbb{E}\left[\|\zeta(\cdot) - \Psi(\cdot, w \odot h)\|_{L^r(\mu)}^q\right]^{\frac{1}{q}} < \epsilon.$$

There also exists a one-hidden-layer node-dropout neural network with the same properties.

Certainly, many variations of the above corollary can be constructed.

To further illustrate Corollary 7, in Figure 4 we look at the approximation in probability of our construction from Theorem 6.

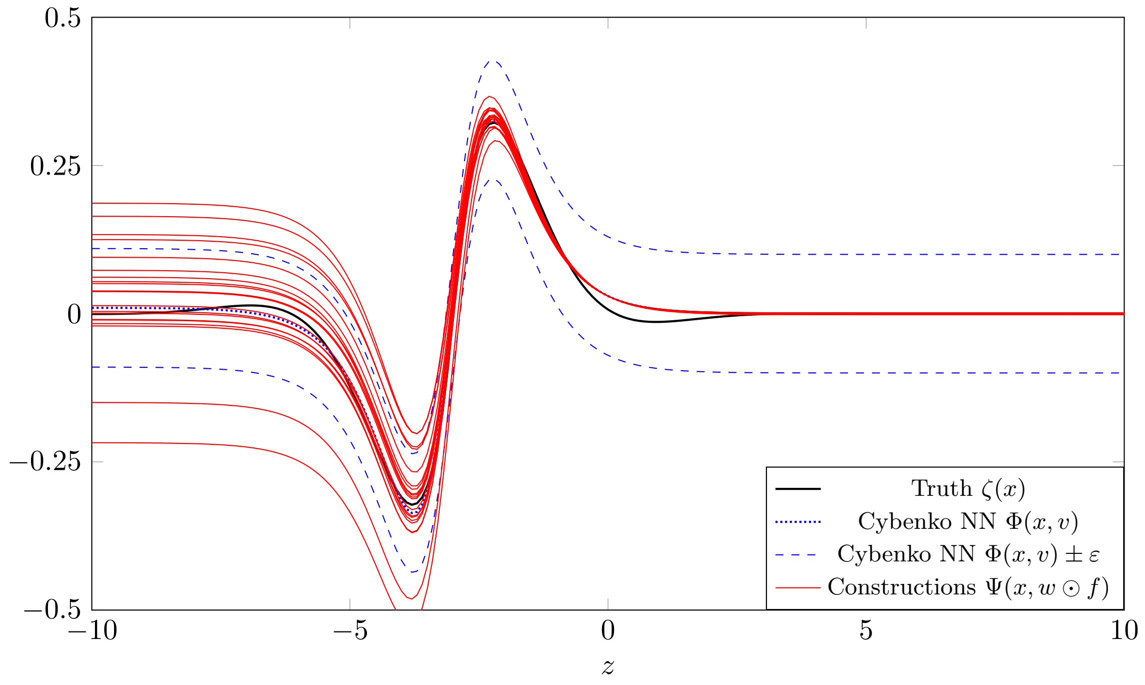

Figure 4: An illustration of function approximation in probability with our construction. Here, $M = 256$, and 20 independent runs of the construction are shown in red. Here, $\epsilon = 0.1$ was chosen for illustrative purposes. Most of the runs lie within ϵ distance around the Cybenko neural network from (15), which we approximate with our construction in (16). Altogether, we are approximating the target function ζ.

3.4 Explicit Computation of Coefficients

To further illustrate Theorem 6, we will compute the coefficients a_U in (13) explicitly for a special case of dropout neural networks for which the filter variables are partitioned into independent blocks. All variables in one block i are all simultaneously off with probability p_i and simultaneously on with probability $1 - p_i$. Both node-dropout and dropconnect are special cases.

Proposition 9 *Let f be a $\{0,1\}^n$-valued random variable with a distribution specified as follows. Let $\overline{1,n} = I_1 \cup \ldots \cup I_r$ be a disjoint partition and suppose that $f_i = f_j$ whenever $i, j \in I_s$ for any $i, j \in \overline{1,n}$ and $s \in \overline{1,r}$. Let $f = (f_{I_1}, \ldots, f_{I_r})$ denote the random variables ordered as blocks and suppose that $\mathbb{P}(f_{I_s} = 1) = 1 - p_s > 0$ for all $s \in \overline{1,r}$ and that $\{f_{I_i}\}_{i\in\overline{1,r}}$ are mutually independent. Then we have*

$$\Psi(\cdot, w) = \sum_{V \in 2^r} \prod_{i \in V} \left(\frac{1}{1-p_i}\right) \prod_{i \in \overline{1,r} \setminus V} \left(-\frac{p_i}{1-p_i}\right) \mathbb{E}\Big[\Psi(\cdot, (w \odot \mathbf{1}_{\iota(V)}) \odot f^V)\Big]$$

where $\iota : 2^r \to 2^n$ is the embedding characterized by $j \in \iota(V)$ if $j \in I_i$ for some $i \in V$.

We prove Proposition 9 in Appendix A.3. Note that as $p_i \to 1$, the coefficients a_U become large. From this fact, together with the observation that the sum is taken over the large set 2^r, it is clear that the construction is computationally strenuous. Still, small examples in the case of dropconnect are shown in Figures 2, 3 and 4.

3.5 Why the Results in this Section are only for Random-approximation Dropout

In this section, we have shown a random-approximation universal approximation result, i.e., a universal approximation result that is relevant when the dropout neural network is also used at prediction time with a stochastic output. In practice, the filter variables are usually replaced by their average values at prediction time. The following example shows that the construction in this section can lead to a bad approximation when doing expectation-replacement.

Example 5 *Let σ be the standard ReLU activation function. The approximation procedure in Corollary 7 would yield that the function $\zeta : \mathbb{R} \to \mathbb{R}$ given by $\zeta(x) := \sigma(x-1)$ can be well approximated by an average of many independent copies of the dropout neural network*

$$x \mapsto 4 f_1 \sigma(f_2 x - 1)$$

where f_1 and f_2 are i.i.d. Bernoulli random variables with success probability $1/2$. However, replacing f_1 and f_2 by $1/2$, we just obtain the function

$$x \mapsto 2\sigma(x/2 - 1)$$

which is not a good approximation to the function ζ at all.

4. Use of Average Filter Variables for Prediction

We will now approximate a neural network by a larger dropout neural network that is also close to the original neural network if the filter variables are replaced by their expected values. The replacement of the filter random variables f by their expected values $\mathbb{E}[f]$ is common practice after having trained dropout neural networks for prediction. Informally, the main Theorem 17 below states that for any base neural network $\Psi(\cdot, w)$, there exists a larger neural network $\mathsf{NN}_{\Gamma,\Xi}(\cdot, v)$ and filter variables f such that

$$\mathsf{NN}_{\Gamma,\Xi}(x, v \odot f) \approx \Psi(x, w) \approx \mathsf{NN}_{\Gamma,\Xi}(x, v \odot \mathbb{E}[f]). \tag{21}$$

Global Variables

In order to improve readability of this section, we fix for the entire section a few (otherwise arbitrary) variables. Throughout this section:

- The base neural network Ψ is assumed to be a fixed $(L-1)$-hidden layer neural network as described in Section 2.1. We assume that its activation functions σ_j are continuous. We also keep the weights $W^{(j)}$ and biases $b^{(j)}$ fixed.
- We fix a number $R > 0$, which will play the role of the radius of a ball in the input space.
- We fix a number $\beta \in (0,1)$ and assume that for every random filter matrix F in this section, each one entry is on with a probability that is larger than or equal to β, i.e., for all r, c,
$$\mathbb{P}[F_{rc} = 1] \geq \beta > 0.$$
- We fix a number $Q > 1$, whose role will become clear later.

4.1 Heuristic Description of the Construction

In this section we describe the construction of the larger dropout neural network $\mathsf{NN}_{\Gamma,\Xi}$ in heuristic terms; the full details are given in the subsequent sections. The construction starts at the last layer of the base neural network Ψ, which is a function $\Psi_L : \mathbb{R}^{d_{L-1}} \to \mathbb{R}^{d_L}$ given by

$$x \mapsto \sigma_L\big(W^{(L)}x + b^{(L)}\big). \tag{22}$$

We construct the last layer of the larger dropout neural network such that it remains close to (22) as follows. Let $N \in \mathbb{N}$ and consider any collection $\{F^{(L),i}\}_{i\in\overline{1,N}}$ of i.i.d. random filter matrices such that for each i, $F^{(L),i}$ has the same dimension as $W^{(L)}$. By the law of large numbers, we can expect that the function

$$x \mapsto \sigma_L\bigg(\frac{1}{N}\sum_{i=1}^{N}\Big(\big(W^{(L)} \div \mathbb{E}[F^{(L),i}]\big) \odot F^{(L),i}\Big)x + b^{(L)}\bigg) \tag{23}$$

will be close to the function Ψ_L for sufficiently large N. Here we write $\div$ for element-wise division.

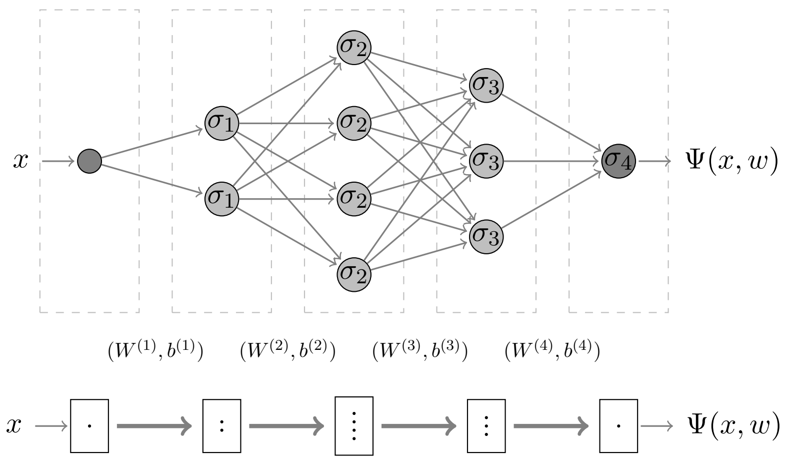

Figure 5: An example base neural network Ψ where $L = 4$. The top diagram indicates the individual activation functions and the dimensions of each layer of nodes: $d = d_0 = 1$, $d_1 = 2$, $d_2 = 4$, $d_3 = 3$, and $d_4 = 1$. The set of all arrows connecting layer $k-1$ of nodes to layer k correspond to the parameters $(W^{(k)}, b^{(k)})$. The bottom diagram shows the same network, with the edges between layers compressed to single arrows; this compressed notation is the basis for the diagram in Figure 6 below.

Viewed as a one-layer neural network, the function (23) is a one-layer dropout neural network with N times as many edges as Ψ_L, and it can replace (22), i.e., Ψ_L, while staying close to Ψ_L.

A further adaptation is necessary, however, because in (23) each copy $W^{(L)} \div \mathbb{E}[F^{(L),i}]$ takes the *same* input $x \in \mathbb{R}^{d_{L-1}}$. To make (23) a *bona fides* dropout network, different edges should take different inputs, and therefore we generalize (23) to

$$(\mathbb{R}^{d_{L-1}})^N \ni (x^i)_{i\in\overline{1,N}} \mapsto \sigma_L\left(\frac{1}{N}\sum_{i=1}^{N}\Big(\big(W^{(L)} \div \mathbb{E}[F^{(L),i}]\big) \odot F^{(L),i}\Big)x^i + b^{(L)}\right). \tag{24}$$

By precomposing each of the inputs x^i with $\Psi^{(L-1)}$, and performing the same construction as above (copying the input to these copies of $\Psi^{(L-1)}$), we can inductively create our larger dropout neural network $\mathsf{NN}_{\Gamma,\Xi}$ that will be close to Ψ.

There are now three points of attention:

- The intuitive statement 'repeating this construction' needs a formalization by an inductive construction. This requires a mathematical object that can record the intermediate stages of the construction.
- We need to show inductively that the resulting intermediate neural networks are close to (a network closely related to) the original network. In particular, we need to introduce a mathematical specification of 'close' that is compatible with an inductive argument.

- The input space to the neural network in this construction grows with each step, whereas we still aim to have a final neural network with data space $\mathbb{R}^d$. This requires us to deal with the first layer of the network differently.

These points are the topics of the subsequent sections.

4.2 Dropout-trees

We will encode the intermediate stages of our inductive construction by a mathematical object that we will refer to as a *dropout-tree.* The idea is that we start with a root, then attach incoming edges labeled with random filter matrices to it (creating leaves), and then recursively attach even more edges to the leaves. To be consistent with the numbering of layers in Section 2.1, here, we will prefer to speak about the *level* j of a vertex or an edge in a tree rather than its depth $L-j$ (the latter is also established jargon in graph theory, and this aligns our notation with that of the neural network). In this numbering, the root is therefore at level L.

Definition 10 *A vertex v of a rooted tree is at* level $j \in \{0, 1, \ldots, L\}$ *if the path from v to the root v_0 has length $L-j$. An edge $e = (u, v)$ of a rooted tree is at* level $j \in \{0, 1, \ldots, L\}$ *if its target vertex v is at level j.*

From now on we will write σ_v for $\sigma_{\mathsf{level}(v)}$, W^v for $W^{(\mathsf{level}(v))}$, b^v for $b^{(\mathsf{level}(v))}$, *et cetera.* This simplifies the notation at only a minor cost of abuse of notation.

Definition 11 *A* dropout-tree Γ *of an $(L-1)$-hidden layer neural network Ψ is a directed graph $\mathcal{G}$ together with a labeling of the edges such that:*

- *the graph $\mathcal{G}$ is connected and acyclic;*
- *one of the vertices, say v_0, is designated as the* root*;*
- *the depth of the tree is at most $L-1$;*
- *all directed edges point towards the root;*
- *every edge e is labeled with a random matrix F^e; for each e*
  - *(a) F^e has the same dimension as $W^{\mathsf{target}(e)}$*
  - *(b) F^e's entries are $\{0,1\}$-valued*
  - *(c) for all r, c, $\mathbb{P}[F^e_{rc} = 1] \geq \beta > 0$;*
- *for every vertex v that is not a leaf, $\{F^e\}_{e \in \mathsf{into}(v)}$ is a collection of mutually independent, identically distributed random matrices.*

For convenience we recall some terminology. A directed edge points from a *source* to a *target*, and for an edge e we identify them by $\mathsf{source}(e)$ and $\mathsf{target}(e)$; we write $\mathsf{into}(v)$ for the set of all edges with target vertex v. In the trees in this paper, all edges point towards the root of the tree. A vertex v is a *child* of a vertex w if there is an edge pointing from v to w; w then is the *parent* of v. A *leaf* is a vertex without children.

Dropout-trees can be constructed iteratively by starting from the trivial dropout-tree consisting only of a root and then performing a so-called *μ-input-copy* construction. This allows us to inductively create a larger dropout-tree from a smaller dropout-tree.

Definition 12 *Let Γ be a dropout-tree and let ℓ be a leaf of Γ at level k. Let μ be a distribution of a random matrix $F \in \{0,1\}^{d_k \times d_{k-1}}$ that satisfies for all r, c, $\mathbb{P}[F_{rc} = 1] \geq \beta > 0$. A dropout-tree Γ' is a* μ-input-copy to the leaf ℓ of Γ *if one can obtain Γ' from Γ by: (a) attaching child vertices to ℓ, and (b) labeling each edge going into ℓ by an independent copy of F. The* size of a μ-input-copy to ℓ at Γ *refers to the number of children of ℓ in Γ'.*

Let us describe the precise meaning of procedure (b) in Definition 12. For that, it may be useful to recall that random matrices are nothing but measurable functions defined on the probability space $(\Omega, \mathcal{F}_\Omega, \mathbb{P})$. The procedure (b) precisely means that the sigma-algebras generated by the filter variables F^e with $e \in \mathsf{into}(\ell)$ are independent, and that for every $e \in \mathsf{into}(\ell)$ the law of F^e equals μ. In particular, this condition allows for some correlation between filter variables labeling edges in the dropout-tree that do not go into ℓ. Moreover, in general there can be many different dropout trees Γ' that are μ-input-copies of Γ.

We will now describe how a dropout-tree encodes a dropout neural network.

4.2.1 Dropout neural networks encoded by dropout-trees

Each dropout-tree Γ will induce a stochastic function $\Phi_\Gamma^{v_0}$—a dropout neural network—as follows. For any edge $e = (u, v)$ of Γ, let

$$V_\Gamma^e := W^e \div \mathbb{E}[F^e] \tag{25}$$

be rescaled weights for the dropout neural network. We define

$$\Phi_\Gamma^v := \begin{cases} \sigma_v\Big(\frac{1}{\#\mathsf{into}(v)} \sum_{e \in \mathsf{into}(v)} (V^e \odot F^e)\Phi_\Gamma^{\mathsf{source}(e)} + b^v\Big) & \text{if } v \text{ is not a leaf,} \\ \mathrm{Identity}_{\mathbb{R}^{d_v}} & \text{if } v \text{ is a leaf.} \end{cases}$$

Figure 6 illustrates this construction, based on the network Ψ depicted in Figure 5.

4.3 Dropout Neural Networks induced by Dropout-trees are Close to their Deterministic Counterpart

We will give an inductive argument that $\Phi_\Gamma^{v_0}$ is close to $\Phi_{\Gamma_{\mathsf{det}}}^{v_0}$. Here, Γ_{det} denotes the same dropout-tree as Γ except for the fact that we have replaced each and every filter variable deterministically by its expectation. Loosely speaking, the inductive argument implies that dropout neural networks induced by dropout-trees are close to their deterministic counterparts.

As a technical preparation, we define a sequence of radii $R_0, R_1, \ldots, R_L$. The idea is that these provide bounds on the output after applying several layers, no matter the choice of filter variables or weights in the upcoming construction. Given the radius $R > 0$ defined at the start of this section, we set

$$R_0 := \frac{Q}{\beta} R + 1$$

and then choose R_j inductively such that for all $j \in \overline{1,L}$, $x \in B(0, \beta^{-1}\|W^{(j)}\|_{\mathrm{HS}} R_{j-1} + 1)$ it holds that

$$\left|\Psi_j\Big(x, \big(I, b^{(j)}\big)\Big)\right| < R_j - 1. \tag{26}$$

for all $j = \overline{1,L}$, where $\beta \in (0,1)$ and $Q > 1$ were two of the global variables that we defined at the beginning of the section. Here $\|\cdot\|_{\mathrm{HS}}$ denotes the Hilbert–Schmidt norm of a matrix and I denotes an identity matrix of the corresponding size.

We denote the input space to a network induced by a dropout-tree Γ by Inp_Γ. That is, Inp_Γ is the vector space

$$(x^\ell \in \mathbb{R}^{d_{\mathsf{level}(\ell)}} \mid \ell \in \mathsf{leaves}(\Gamma)).$$

We endow Inp_Γ with the norm

$$\|(x^\ell)\|_{\mathsf{Inp}_\Gamma} := \max_{l \in \mathsf{leaves}(\Gamma)} \|x^\ell\|_{\mathbb{R}^{d_{\mathsf{level}(\ell)}}}.$$

We define $\mathsf{In}_\Gamma : \mathbb{R}^d \to \mathsf{Inp}_\Gamma$ to be the collection of functions, indexed by leaves ℓ of Γ, that are generated by those layers in the base network Ψ that are *not* represented in Γ at leaf ℓ:

$$\mathsf{In}_\Gamma^\ell(x) := \Big(\Psi_{\mathsf{level}(\ell)}(\cdot, (W^{(\mathsf{level}(\ell))}, b^{(\mathsf{level}(\ell))})) \circ \cdots \circ \Psi_1(\cdot, (W^{(1)}, b^{(1)}))\Big)(x). \tag{27}$$

Note that by the definitions (26) of the radii R_j we have

$$\mathsf{In}_\Gamma^\ell\big(\overline{B(0,R)}\big) \subset B(0, R_{\mathsf{level}(\ell)} - 1). \tag{28}$$

We say that a dropout-tree Γ satisfies property $\mathsf{ApProp}_\Gamma(\delta, \epsilon)$ if

$$\mathbb{P}\Big[\sup_{x \in \overline{B(0,R)}} \sup_{\tilde{x} \in \overline{B(\mathsf{In}_\Gamma(x), \delta)}} \big|\Phi_\Gamma^{v_0}(\tilde{x}) - \Phi_{\Gamma_{\det}}^{v_0}(\mathsf{In}_\Gamma(x))\big| > \frac{\epsilon}{2}\Big] < \Big(\frac{\epsilon}{4R_L}\Big)^q. \tag{29}$$

We will prove Lemma 13: its message is that one can always construct a full dropout-tree that satisfies $\mathsf{ApProp}_\Gamma(\delta, \epsilon)$ for some $\delta > 0$, by copying inputs at vertices.

Lemma 13 *Let Γ be a dropout-tree and let ℓ be a leaf of Γ at level $k > 1$. Let μ be the distribution of a random matrix $F \in \{0,1\}^{d_k \times d_{k-1}}$ that satisfies for all r, c, $\mathbb{P}[F_{rc} = 1] \geq \beta > 0$. The following now holds: if Γ satisfies $\mathsf{ApProp}_\Gamma(\delta, \epsilon)$ in (29) for some $\delta, \epsilon > 0$, then for every sufficiently large μ-input-copy Γ' at ℓ there exists a $\delta' > 0$ such that Γ' satisfies $\mathsf{ApProp}(\Gamma')(\delta', \epsilon)$.*

The proof of Lemma 13 is relegated to Appendix B.1. There, we show that Lemma 13 follows from Lemma 14, which is displayed next and proved in Appendix B.2.

Lemma 14 *Consider any continuous function $\sigma : \mathbb{R}^m \to \mathbb{R}^m$ and let $W \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$. Let $\{F^i\}_{i \geq 1}$ be a sequence of mutually independent copies of a random matrix $F \in \mathbb{R}^{m \times n}$ that satisfies: for $r \in \overline{1,m}$ and $c \in \overline{1,n}$, $0 < \mathbb{E}[F_{rc}] < \infty$ and $0 \leq F_{rc} \leq M < \infty$ w.p. one. Let $V := W \div \mathbb{E}[F]$. The following now holds: for every $0 \leq K < \infty$ and $\rho > 0$ there exists a $\delta > 0$ such that*

$$\mathbb{P}\Big[\sup_{x \in \overline{B(0,K)}} \sup_{(\tilde{x}^i) \in \overline{B(x,\delta)}^N} \Big|\sigma\Big(\frac{1}{N}\sum_{i=1}^N (V \odot F^i)\tilde{x}^i + b\Big) - \sigma(Wx + b)\Big| > \rho\Big] \to 0 \tag{30}$$

as $N \to \infty$.

4.4 Replacing the First Layer

We look now at the first layer and consider first, the case that we do not use dropout and secondly, the case when we do.

4.4.1 Copying the First Layer many times, without Dropout of Input Edges

We will now start from a full dropout-tree and connect copies of the first layer many times to obtain a neural network that approximates the original neural network well, both in terms of random approximation and in terms of expectation replacement. In this section, it is crucial that there is no dropout of edges in this very first layer.

Assume therefore that we have constructed a full dropout tree, i.e., a dropout tree of which all leaves are at level 1 (at depth $L-1$). We denote by $\mathsf{NN}_{\Gamma,\Xi}$ the neural network that is formed by precomposing every leaf ℓ of the neural network $\Phi_\Gamma^{v_0}$ induced by the dropout tree by the first layer of the original network. To align with the next section, we denote this function by $\Xi^\ell := \Phi^{(1)}(\cdot, (W^{(1)}, b^{(1)}))$. We also let $\mathsf{NN}_{\Gamma,\Xi}^{\mathsf{avg-filt}}$ denote almost the same network, but with the modification that every random filter variable is replaced by its expectation.

The next theorem expresses that this construction yields a dropout neural network that approximates the original neural network well in terms of random approximation and expectation replacement.

Theorem 15 *Fix $0 < \epsilon < 1$. Let Γ be a full dropout-tree satisfying $\mathsf{ApProp}_\Gamma(\delta, \epsilon)$ for some $\delta > 0$. Let $\mathsf{NN}_{\Gamma,\Xi}$ and $\mathsf{NN}_{\Gamma,\Xi}^{\mathsf{avg-filt}}$ be the networks describe above, or more precisely defined in Example 6. Then*

$$\mathbb{P}\Big[\sup_{x\in\overline{B(0,R)}} \big|\mathsf{NN}_{\Gamma,\Xi}(x) - \Psi(x,w)\big| > \epsilon\Big] < \epsilon \tag{31}$$

and

$$\mathbb{E}\Big[\sup_{x\in\overline{B(0,R)}} \big|\mathsf{NN}_{\Gamma,\Xi}(x) - \Psi(x,w)\big|^q\Big]^{1/q} < \epsilon, \tag{32}$$

while

$$\sup_{x\in\overline{B(0,R)}} \Big|\mathsf{NN}_{\Gamma,\Xi}^{\mathsf{avg-filt}}(x) - \Psi(x,w)\Big| < \epsilon. \tag{33}$$

The theorem follows from Theorem 17, which will deal with the more general case in which edges in the input layers can be dropped. Indeed, if in Theorem 17 one takes $\sigma_0 : \mathbb{R} \to \mathbb{R}$ to be the identity function and one sets all filter variables deterministically equal to 1, then for every α and N, the networks constructed in Theorem 17 coincide exactly with $\mathsf{NN}_{\Gamma,\Xi}$ and $\mathsf{NN}_{\Gamma,\Xi}^{\mathsf{avg-filt}}$ introduced above.

4.4.2 First Layers in which Inputs are Dropped

The situation is more complicated if we allow for dropout of edges in every layer, particularly the first. In this section we will show that we can also in that case find a dropout neural network that approximates the original neural network well both in terms of random approximation and in terms of expectation replacement.

Assume again that we have constructed a full dropout-tree, that is, a dropout-tree of which all leaves are at level 1 (i.e., at depth $L-1$). This means that we have constructed suitable replacements for almost every layer of the neural network, except for the first layer. This layer contains the edges that have the global input as source. Replacing the first layer requires a different construction: if we would outright drop edges in the first layer, then we can not control the error with the current technique. We now describe how we replace the first layer.

For every leaf ℓ in the full dropout-tree, we precompose every input at ℓ with a stochastic function $\Xi^{\ell} : \mathbb{R}^{d_0} \to \mathbb{R}^{d_1}$. We record this information in what we call a *precomposition* for a dropout-tree. Figure 6 illustrates this precomposition.

Definition 16 *A* precomposition Ξ *for a full dropout-tree* Γ *is a map* $\Xi : \mathsf{leaves}(\Gamma) \to (\mathbb{R}^{d_0} \to \mathbb{R}^{d_1})$ *that sends every leaf* $\ell \in \mathsf{leaves}(\Gamma)$ *to a stochastic function* $\Xi^{\ell} : \mathbb{R}^{d_0} \to \mathbb{R}^{d_1}$.

Let $\Delta : \mathbb{R}^{d_0} \to (\mathbb{R}^{d_0})^{\mathsf{leaves}_\Gamma}$ be the diagonal map sending x to copies of x. The neural network induced by the full dropout-tree Γ and a precomposition Ξ that we consider is given by

$$\mathsf{NN}_{\Gamma,\Xi} := \Phi^{v_0}_{\Gamma,\Xi} \circ \Delta \tag{34}$$

where

$$\Phi^{v}_{\Gamma,\Xi} = \begin{cases} \sigma_v\Big(\frac{1}{\#\mathsf{into}(v)} \sum_{e\in\mathsf{into}(v)} (V^e \odot F^e)\Phi^{\mathsf{source}(e)}_{\Gamma,\Xi} + b^e\Big) & \text{if } v \text{ is not a leaf,} \\ \Xi^v & \text{if } v \text{ is a leaf.} \end{cases} \tag{35}$$

We also define $\mathsf{NN}^{\mathsf{avg-filt}}_{\Gamma,\Xi}$ as being almost the same neural network as $\mathsf{NN}_{\Gamma,\Xi}$, with the only difference being that we replace each random filter variable F^e in (35) with its expectation $\mathbb{E}[F^e]$. Recall for (34) that v_0 designates the root of the dropout-tree, and note that (35) constructs the neural network recursively (layer by layer).

Example 6 *In fact, we already examined one specific precomposition* Ξ *in* Section 4.4.1*: the one that assigns the function* $\Psi^1(\cdot,(W^{(1)},b^{(1)}))$ *to every leaf. This precomposition yields a neural network* $\mathsf{NN}_{\Gamma,\Xi}$ *in which edges in the first layer, i.e., the input edges of the neural network, are never dropped. In this case,* $\mathsf{NN}^{\mathsf{avg-filt}}_{\Gamma,\Xi}(w)$ *actually coincides with the original neural network* $\Psi(\cdot,w)$.

We will now construct precompositions that allow for the possibility of dropping edges in the first layer and applying e.g., the ReLU function to them immediately. Concretely, we will add a zeroth layer with an activation function $\sigma_0 : \mathbb{R} \to \mathbb{R}$. We assume that $\sigma_0(0) = 0$ and that σ_0 has one-sided derivatives σ_- and σ_+ in the point $0 \in \mathbb{R}$:

$$\sigma_- := \lim_{\alpha\downarrow 0} \frac{\sigma_0(-\alpha) - \sigma_0(0)}{\alpha}, \quad \sigma_+ := \lim_{\alpha\downarrow 0} \frac{\sigma_0(+\alpha) - \sigma_0(0)}{\alpha}. \tag{36}$$

Define the sign function

$$S(x) = \begin{cases} - & \text{if } x < 0, \\ + & \text{if } x \geq 0 \end{cases} \tag{37}$$

component-wise. If $x \neq 0$, then $\sigma_{S(-x)} + \sigma_{S(x)} = \sigma_- + \sigma_+$ does not depend on x—a critical fact that we will leverage in our construction.

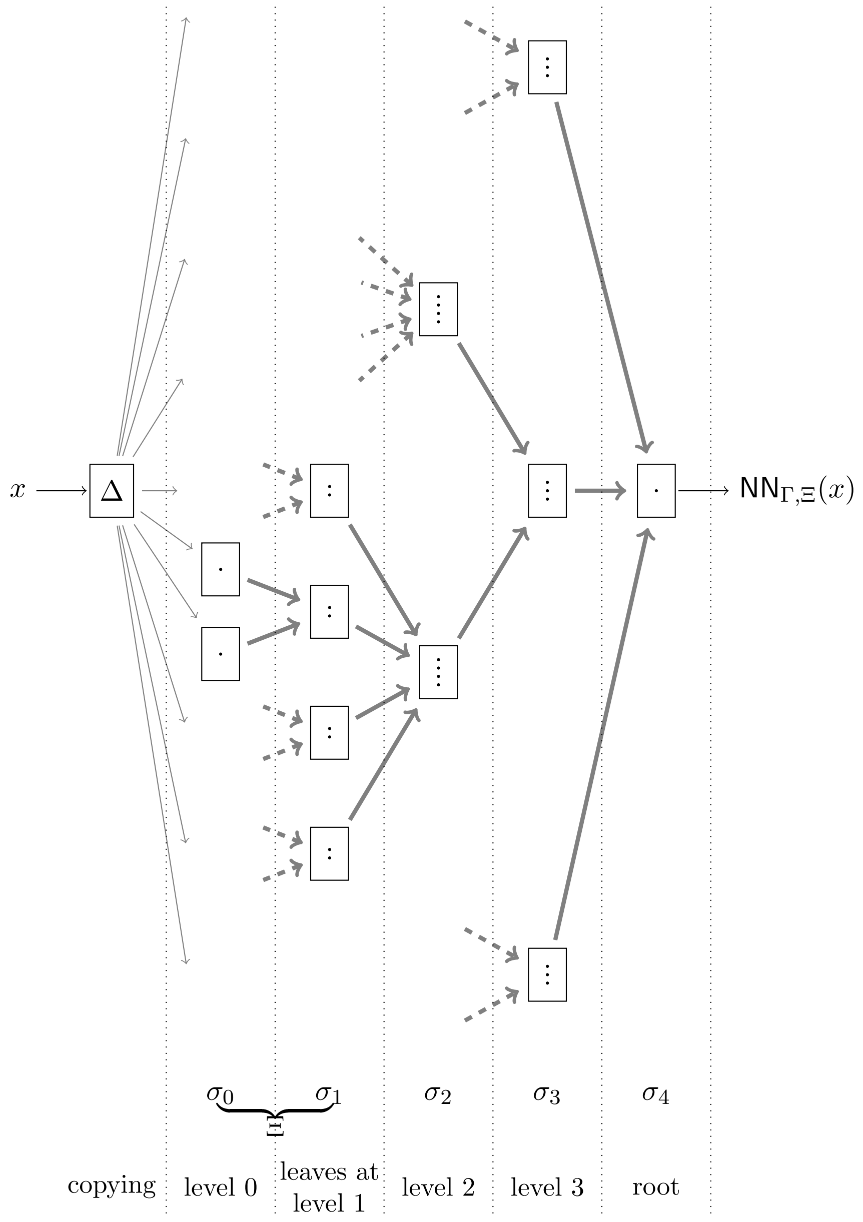

Figure 6: An illustration of how we use the base neural network Ψ from Figure 5 and a dropout-tree (indicated with thicker arrows) to ultimately construct the larger neural network $\mathsf{NN}_{\Gamma,\Xi}$ in which edges are being dropped stochastically. Here, $L = 4$.

Example 7 *Consider a zeroth layer that is the identity function, i.e., $\sigma_0(y) = y$. Then $\sigma_\pm = 1$. Choosing σ_0 as the identity function is allowed here, and means that the layer is not adapted.*

Example 8 *Consider a zeroth layer with ReLU activation function, $\sigma_0 := \mathrm{ReLU}(z) := z\mathbb{1}[z \geq 0]$. Then $\sigma_- = 0$ and $\sigma_+ = 1$.*

Here are the precompositions that we employ: we call Ξ an (α, N)-*precomposition associated to a set of distributions* $\{\mu^\ell, \nu^\ell\}_{l\in\mathrm{leaves}(\Gamma)}$ if for each leaf ℓ,

$$\Xi^\ell(x) := \sigma_1\Big(\frac{1}{N}\sum_{i=1}^{2N}(-1)^i(V^\ell \odot F^{\ell,i})\sigma_0\big((-1)^i\alpha(I \odot G^{\ell,i})x\big) + b^\ell\Big) \tag{38}$$

where element-wise

$$V^\ell_{rc} = \frac{W^{(1)}_{rc}}{\alpha\big(\sigma_- + \sigma_+\big)\,\mathbb{E}[F^\ell_{rc}]\,\mathbb{E}[G^\ell_{cc}]}, \tag{39}$$

and $\{F^{\ell,i}\}_{i\geq 1}$, $\{G^{\ell,i}\}_{i\geq 1}$ are sequences of mutually independent copies of random matrices F^ℓ, G^ℓ that have distributions μ^ℓ, ν^ℓ, respectively. Furthermore, the F^ℓ are presumed to have the same size as $W^{(1)}$, and the G^ℓ to have size $d_0 \times d_0$. Note that these assumptions allow us to place unit mass on any particular outcome and thus to replace F^ℓ, G^ℓ by deterministic counterparts.

The idea of (38) is that it represents two layers of a dropout neural network that satisfies the approximation properties we are after. The functions σ_1, σ_0 can be understood as their activation functions, the matrices V^ℓ, αI as their weights, b^ℓ as a bias, and the matrices F^ℓ, G^ℓ as describing which edges and inputs are randomly removed. By scaling the weights V^ℓ by $1/(2N)$ and generating $2N$ independent copies of the first layer, we are preparing for an application of the law of large numbers. Furthermore, by allowing for arbitrarily small α, we are preparing for a linearization of σ_0 around 0. Finally, the alternatingly positive and negative multiplicative factors $(-1)^i$ allow us to cover directional derivatives such as that of the ReLU activation function. All together, the construction allows us to prove the following theorem.

Theorem 17 *Fix $0 < \epsilon < 1$. Let Γ be a full dropout-tree satisfying $\mathsf{ApProp}_\Gamma(\delta, \epsilon)$ for some $\delta > 0$. Let Ξ be an (α, N)-precomposition associated to a set of distributions $\{\mu^\ell, \nu^\ell\}_{\ell\in\mathrm{leaves}(\Gamma)}$. Assume that for every ℓ, if F is a matrix of filter variables distributing according to μ^ℓ or ν^ℓ, then for every r, c,*

$$\mathbb{P}[F_{rc} = 1] \geq \beta > 0.$$

Let $\sigma_0 : \mathbb{R} \to \mathbb{R}$ be a continuous function with one-sided derivatives σ_- and σ_+ in 0, such that $\sigma_- + \sigma_+ \neq 0$ and such that $\sigma_0(0) = 0$. Assume moreover that σ_- and σ_+ satisfy the following inequality with respect to the global variable Q:

$$4\frac{|\sigma_-| + |\sigma_+|}{|\sigma_- + \sigma_+|} < Q. \tag{40}$$

The following inequalities now hold for $\alpha > 0$ small enough and $N \in \mathbb{N}$ large enough:

$$\mathbb{P}\Big[\sup_{x\in\overline{B(0,R)}}\big|\mathsf{NN}_{\Gamma,\Xi}(x) - \Psi(x,w)\big| > \epsilon\Big] < \epsilon \tag{41}$$

and

$$\mathbb{E}\Big[\sup_{x\in\overline{B(0,R)}}\big|\mathsf{NN}_{\Gamma,\Xi}(x) - \Psi(x,w)\big|^q\Big]^{1/q} < \epsilon, \tag{42}$$

while

$$\sup_{x\in\overline{B(0,R)}}\Big|\mathsf{NN}^{\mathsf{avg-filt}}_{\Gamma,\Xi}(x) - \Psi(x,w)\Big| < \epsilon. \tag{43}$$

An important consequence of Theorem 17 is that we obtain for instance a universal approximation result for node-dropout and dropconnect neural networks with ReLU activation functions that also guarantees a good approximation when filter variables are replaced by their averages, as formalized by Corollaries 4 and 5 in the introduction. Theorem 17 is proven in Appendix B.3. There, we show that Theorem 17 follows from the following Lemma 18, which in turn is proven in Appendix B.4 using compactness arguments and the law of large numbers.

Lemma 18 *Let $\sigma_0 : \mathbb{R} \to \mathbb{R}$ be a continuous function with $\sigma(0) = 0$ and with two one-sided derivatives σ_- and σ_+ in 0 satisfying $|\sigma_- + \sigma_+| > 0$. Let Ξ be an (α, N)-precomposition associated to a set of distributions $\{\mu^\ell, \nu^\ell\}_{l\in\text{leaves}(\Gamma)}$ such that for all ℓ, r, c, $\mathbb{E}[F^\ell_{rc}] > 0, \mathbb{E}[G^\ell_{rc}] > 0$, $0 \le F^\ell_{rc} \le M$ w.p. one, and $0 \le G^\ell_{rc} \le M < \infty$ w.p. one. The following now holds: for every leaf ℓ, $0 \le K < \infty$, and $\rho > 0$, for α small enough and N large enough,*

$$\mathbb{P}\Big[\sup_{x\in\overline{B(0,K)}}\big|\Xi^\ell(x) - \Psi_1(x;(W^{(1)}, b^{(1)}))\big| > \rho\Big] < \rho \tag{44}$$

and

$$\sup_{x\in\overline{B(0,K)}}\big|\Xi^{\ell,\mathsf{avg-filt}}(x) - \Psi_1(x;(W^{(1)}, b^{(1)}))\big| < \rho \tag{45}$$

where $\Xi^{\ell,\mathsf{avg-filt}}$ denotes the function Ξ^ℓ in (38) but with each filter variable $F^{\ell,i}$ replaced by its expectation $\mathbb{E}[F^\ell]$.

5. Discussion

In this article, we showed that dropout neural networks are rich enough for a universal approximation property to hold, both for random-approximation and expectation-replacement dropout. It is further evidence that the representational capacity of neural networks is so large that approximations are possible despite significant additional constraints. In the case of dropout in general, these additional constraints are the implicit symmetry constraints enforced by the turning on and off of the filter variables: in dropconnect, for instance, for most realizations of the filter variables, the output of the dropconnect neural network still approximates the original neural network well after the filter variables are randomly permuted. Despite the enforced invariance with respect to this operation, there is enough room in the parameter space for the weights of the network to have a good approximation for the overwhelming majority of realizations of the filter variables.

Our proof of the universal approximation property for random-approximation dropout explicitly works with this symmetry. By this, we mean the following. The universal approximation property that we show even works when edges from the input nodes are dropped out at random. The output in the first hidden layer is then inherently *random*, and in no way close to deterministic. This is in contrast with for instance the universal approximation property by Foong et al. (2020) in which the layers are all very close to deterministic. Yet even though the values in the nodes are random, we do have a good understanding of the *distribution* of the values in the nodes, and two stochastic realizations are most likely almost permutations of each other. By blowing up the first layer, i.e., repeating it many times in parallel, we then know the output very well up to this permutation symmetry and this turns out to be enough for us to show a universal approximation property.

5.1 Limitations of our Results

Our results and methods have several limitations.

We only show the *existence* of dropout neural networks close to a given function. It is a completely separate question whether an algorithm such as dropout stochastic gradient descent would actually be able to find such an approximation. The main message of our result is that at least there is no theoretical obstruction to approximating functions with dropout neural networks.

In the proofs, we used very explicitly that filter variables only take on the values zero or one, while other forms of dropout also exist (for instance with Gaussian filter variables). Our algebraic proof does not readily generalize to this more general case, but it is possible that parts of the proof could be reused.

In this paper we made no efforts to reduce the number of parameters of the approximating networks, and indeed the number of parameters can rapidly grow with increasing dimensions of the parameter w. This can for instance be recognized in the exponential number of additional parameters a_U in Theorem 6, the large (but algebraic) number M of copies that is required to reduce variance in (14), and the exponential increase of leaves in dropout-trees with increasing depth. Naturally, understanding optimal approximation rates in terms of parameter dimensions is a central question in Approximation Theory, but we leave this to future work.

6. Conclusion

We showed two types of universal approximation results for dropout neural networks, one for random-approximation dropout, in which case the random filter variables are also used at prediction time, and one for expectation-replacement dropout, in which case the filter variables are replaced by their averages at prediction time. Our results allow for dropout of edges from the input layer, allow for a wide class of distributions on filter variables, including dropout of edges from the input layer, and for a wide class of activation functions.

By making the difference between random-approximation and expectation-replacement dropout explicit, our results also highlight the following mystery: How is it that expectation-replacement dropout performs so well on prediction time?

Acknowledgments

J.W. Portegies was supported by the Electronic Component Systems for European Leadership Joint Undertaking under grant agreement No 737459 (project Productive 4.0). This Joint Undertaking receives support from the European Union Horizon 2020 research and innovation program and Germany, Austria, France, Czech Republic, Netherlands, Belgium, Spain, Greece, Sweden, Italy, Ireland, Poland, Hungary, Portugal, Denmark, Finland, Luxembourg, Norway, Turkey.

Appendix A. Proofs of Section 3

In this appendix we prove the results of Section 3. In particular, we prove Theorem 6, Corollary 7 and Proposition 9.

A.1 Proof of Theorem 6

We require the following algebraic lemma, which lies at the heart of Theorem 6, as it implies the existence of the constants (a_U).

Lemma 19 *Let $\Psi : \mathbb{R}^d \times \mathbb{R}^n \to \mathbb{R}$ be a function. Let (f^U) be a collection of $\{0,1\}^n$-valued random variables indexed by subsets $U \subset \overline{1,n}$, such that for every U*

$$\mathbb{P}[f^U = (1,\dots,1)] > 0.$$

Then for every subset $V \subset \overline{1,n}$, it holds that

$$\Psi(\cdot, \cdot \odot \mathbf{1}_V) \in \operatorname{span}\left\{\mathbb{E}\left[\Psi(\cdot, (\cdot \odot \mathbf{1}_U) \odot f^U)\right] : U \in 2^n\right\}$$

where for any subset $S \in 2^d$, i.e., $S \subset \{1,\dots,d\}$, we denote by $\mathbf{1}_S$ the characteristic function of S.

In other words, there exist constants $(a_{U,V})_{U \in 2^n}$ independent of w and x such that for all $(x,w) \in \mathbb{R}^d \times \mathbb{R}^n$

$$\Psi(x, w \odot \mathbf{1}_V) = \mathbb{E}\left[\sum_{U \in 2^n} a_{U,V} \Psi(x, (w \odot \mathbf{1}_U) \odot f^U)\right],$$

and in particular there exists constants α_U such that

$$\Psi(x,w) = \mathbb{E}\left[\sum_{U \in 2^n} a_U \Psi(x, (w \odot \mathbf{1}_U) \odot f^U)\right].$$

Proof The proof is by induction on the cardinality of V and follows from the equality

$$\Big(\sum_{S:V \subset S} \mathbb{P}[f^V = \mathbf{1}_S]\Big)\Psi(\cdot, w \odot \mathbf{1}_V) = \mathbb{E}\Big[\Psi(\cdot, (w \odot \mathbf{1}_V) \odot f^V)\Big] - \sum_{S:V \setminus S \neq \emptyset} \mathbb{P}[f^V = \mathbf{1}_S]\Psi(\cdot, w \odot \mathbf{1}_{S \cap V}). \qquad (46)$$

In particular, for the base case in which V is empty, the last term vanishes. In the induction step, the functions $\Psi(\cdot, w \odot \mathbf{1}_{S\cap V})$ are by the induction hypothesis all in the required span. ■

Proof (of Theorem 6) By Lemma 19, we can find constants a_U for $U \in 2^n$ such that (13) holds. We look now at (14). By the law of large numbers, convergence in probability in the normed vector space $(\mathcal{F}, \|\cdot\|)$ follows. Moreover, for any $V \in 2^n$ we have

$$\|\Psi(\cdot, (w \odot \mathbf{1}_V) \odot f^V)\|_{\mathcal{F}} \le \max_{U\in 2^n} \|\Psi(\cdot, (w \odot \mathbf{1}_U) \odot f^U)\|_{\mathcal{F}} =: C_w, \tag{47}$$

so that for any $q \in [1, \infty)$ and M,

$$\mathbb{E}\Big[\Big\|\frac{1}{M}\sum_{i=1}^{M}\sum_{U\in 2^n} a_U \Psi(\cdot, (w \odot \mathbf{1}_U) \odot f^{i,U})\Big\|_{\mathcal{F}}^q\Big]^{\frac{1}{q}} \le \max_{U\in 2^n} |a_U| C_w. \tag{48}$$

With uniform boundedness for all M, we can use the dominated convergence theorem which implies then convergence in L^q of the $\mathcal{F}$-valued random variables as $M \to \infty$. ■

A.2 Proof of Corollary 7

Proof Let $\zeta \in \mathcal{F}$ and let $\epsilon > 0$. Assume there exists a $(m, \Phi, f) \in \mathsf{DDNN}$ and a $v \in \mathbb{R}^m$ such that $\|\Phi(\cdot, v) - \zeta\|_{\mathcal{F}} < \epsilon$. Define $\eta := \epsilon - \|\Phi(\cdot, v) - \zeta\|_{\mathcal{F}} > 0$. Define the collection (f^U) of $\{0,1\}^m$-valued filter variables, each being specifically an independent copy of f. By Theorem 6, there exist constants (a_U), a number $M \in \mathbb{N}$ and $2^m M$ independent copies $(f^{i,U})$ of f such that

$$\mathbb{P}\Big[\Big\|\frac{1}{M}\sum_{i=1}^{M}\sum_{U\in 2^m} a_U \Phi(\cdot, (v \odot \mathbf{1}_U) \odot f^{i,U}) - \Phi(\cdot, v)\Big\|_{\mathcal{F}} > \eta\Big] < \eta$$

and

$$\mathbb{E}\Big[\Big\|\frac{1}{M}\sum_{i=1}^{M}\sum_{U\in 2^m} a_U \Phi(\cdot, (v \odot \mathbf{1}_U) \odot f^{i,U}) - \Phi(\cdot, v)\Big\|_{\mathcal{F}}^q\Big]^{1/q} < \eta.$$

Hence by the triangle inequality, in fact

$$\mathbb{P}\Big[\Big\|\frac{1}{M}\sum_{i=1}^{M}\sum_{U\in 2^m} a_U \Phi(\cdot, (v \odot \mathbf{1}_U) \odot f^{i,U}) - \zeta\Big\|_{\mathcal{F}} > \epsilon\Big] < \epsilon$$

and

$$\mathbb{E}\Big[\Big\|\frac{1}{M}\sum_{i=1}^{M}\sum_{U\in 2^m} a_U \Phi(\cdot, (v \odot \mathbf{1}_U) \odot f^{i,U}) - \zeta\Big\|_{\mathcal{F}}^q\Big]^{1/q} < \epsilon.$$

We then define the tuple $(n, \Psi, g) \in \mathsf{DDNN}$ as an independent finite linear combination of $2^m M$ copies of (m, Φ, f), with coefficients a_U/M. Setting $\tilde{v} \in \mathbb{R}^{2^m m}$ to be the concatenation of the 2^m modified vectors $(v \odot \mathbf{1}_U)_{U\subset\overline{1,m}}$, we then set $w \in \mathbb{R}^{2^m M m}$ to be the subsequent concatenation of M copies of $\tilde{v}$. The combination of (n, Ψ, g) and w achieves the assertion. ■

A.3 Proof of Proposition 9

As we have seen in Lemma 19, we can find the map $\Psi(\cdot, w)$ in the span of $\mathbb{E}[\Psi(\cdot, (w \odot \mathbf{1}_V) \odot f^V]$ for $V \in 2^n$. We will look now at specific cases where we can explicitly compute the linear combination. In particular we will look in the case that we use *dropout* (Hinton et al., 2012), that is, we drop nodes independently with the same probability, and *dropconnect* (Wan et al., 2013), where we drop individual weights independently with the same probability.

In both cases the filter variables f_i take the same values for some disjoint subsets of $\overline{1,n}$, where n is the number of weights where we apply filters. That is, if we have a disjoint set decomposition $\overline{1,n} = I_1 \cup \ldots \cup I_r$ with $I_k \cap I_s = \emptyset$ whenever $k \neq s$, then $f_i = f_j$ for all $i, j \in I_k$ and f_i, f_l are independent if they belong to disjoint sets, $f_i \in I_k$ and $f_l \in I_s$ with $s \neq k$. In this section we drop the index $U \in 2^n$ of the random variable f^U for notational convenience as they are identically distributed. We will use this property to obtain an explicit decomposition in (13) and in a more general setting where the probability of the filters may differ depending on which disjoint set they belong to. For $K, S \in 2^n$, we denote $K \subseteq S$ to be the usual set inclusion, that is, $i \notin K$ whenever $i \notin S$ for all $i \in \overline{1,n}$ holds.

We need the following lemmas:

Lemma 20 *Let $1 \geq q_1, \ldots, q_r > 0$ and $r \in \mathbb{N}$. For $K \in 2^r$,*

$$\mu_K := \sum_{S : K \subseteq S} \prod_{i \in S} q_i \prod_{i \in \overline{1,r} \setminus S} (1 - q_i) \prod_{i \in K} \Big(\frac{1}{q_i}\Big) \prod_{i \in S \setminus K} \Big(1 - \frac{1}{q_i}\Big) \tag{49}$$

satisfies

$$\mu_K = \begin{cases} 1 & \text{if } K = \overline{1,r} \\ 0 & \text{otherwise.} \end{cases} \tag{50}$$

Proof Let $x_1, \ldots, x_r$ be free variables. For $S \in 2^r$ we denote the monomial $x^S = \prod_{i \in S} x_i$. We will prove the identity by comparing coefficients of two equal polynomials. We have

$$q(x_1, \ldots, x_r) = \prod_{i=1}^{r} (q_i x_i + (1 - q_i)) = \sum_{S \in 2^r} x^S \prod_{i \in S} q_i \prod_{i \in \overline{1,r} \setminus S} (1 - q_i) \tag{51}$$

where we have expanded all $|2^r|$ monomials appearing in the decomposition of q. Now we set $x_i = (1/q_i) y_i - (1 - q_i)/q_i$ in (51). We have

$$\begin{aligned} q\Big(\frac{1}{q_1} y_1 - \frac{1 - q_1}{q_1}, \ldots, \frac{1}{q_r} y_r - \frac{1 - q_r}{q_r}\Big) &= \prod_{i=1}^{r} \Big(q_i \Big(\frac{1}{q_i} y_i - \frac{1 - q_i}{q_i}\Big) + (1 - q_i)\Big) \\ &= \prod_{i=1}^{r} y_i = y^{\overline{1,r}}. \end{aligned} \tag{52}$$

On the other hand, if we substitute $x_i = (1/q_i)y_i - (1-q_i)/q_i$ in the monomials x^S in (51) we have

$$\begin{aligned}
\sum_{S\in 2^r} x^S \prod_{i\in S} q_i \prod_{i\in\overline{1,r}\setminus S}(1-q_i) &= \sum_{S\in 2^r}\prod_{i\in S}\Big(\frac{1}{q_i}y_i - \frac{1-q_i}{q_i}\Big)q_i \prod_{i\in\overline{1,r}\setminus S}(1-q_i)\\
&= \sum_{S\in 2^r}\sum_{K\in 2^r : K\subseteq S}\prod_{i\in S} q_i \prod_{i\in\overline{1,r}\setminus S}(1-q_i)\prod_{i\in K} y_i\Big(\frac{1}{q_i}\Big)\prod_{i\in S\setminus K}\Big(-\frac{1-q_i}{q_i}\Big)\\
&= \sum_{K\in 2^r} y^K \sum_{S:K\subseteq S}\prod_{i\in S} q_i \prod_{i\in\overline{1,r}\setminus S}(1-q_i)\prod_{i\in K}\Big(\frac{1}{q_i}\Big)\prod_{i\in S\setminus K}\Big(1-\frac{1}{q_i}\Big)\\
&= \sum_{K\in 2^r}\mu_K y^K \qquad (53)
\end{aligned}$$

so that we must have $\mu_K = 1$ if $K = \overline{1,r}$ and zero otherwise. ∎

Let $\overline{1,n} = I_1 \cup \dots, \cup I_r$ be a disjoint partition of $\overline{1,n}$, i.e., $I_j \cap I_i = \emptyset$ if $i \neq j$. We consider $S \in 2^r$ also as an element of 2^n via the inclusion $\iota : 2^r \to 2^n$ given by $j \in \iota(S)$ if $j \in I_i$ and $i \in S$, i.e., we consider the index i as the set of all indices $j \in I_i$. Note then that $\iota(\overline{1,r}) = \overline{1,n}$. Recall now that the filter random variable with values in $\{0,1\}^n$ are denoted by $f = (f_1, \cdots, f_n)$. We suppose now that the filter random variables satisfy that $f_i = f_j$ whenever $i, j \in I_s$ for some $s \in \overline{1,r}$. We denote by B_s the $\{0,1\}$ valued random variable corresponding to the I_s part of $\overline{1,n}$. We suppose that $\mathbb{P}(B_s = 1) = q_s = 1 - p_s$ for all $s \in \overline{1,r}$, where q_s is the probability of success and p_s the dropping probability. Moreover, we suppose that the $(B_s)_{s\in\overline{1,r}}$ are mutually independent. With this notation we have:

$$\mathbb{E}\big[\Psi(\cdot, w\odot f)\big] = \sum_{L\in 2^r}\prod_{i\in L} q_i \prod_{i\in\overline{1,r}\setminus L}(1-q_i)\Psi(\cdot, w\odot \mathbf{1}_{\iota(L)}). \qquad (54)$$

In the following Lemma, we embed 2^r into 2^n as blocks according to a partition of $\overline{1,n}$ using ι:

Lemma 21 *For $S \in 2^r$,*

$$\mathbb{E}\big[\Psi(\cdot, (w\odot \mathbf{1}_{\iota(S)})\odot f)\big] = \sum_{K\in 2^r : K\subseteq S}\prod_{i\in K} q_i \prod_{i\in S\setminus K}(1-q_i)\Psi\left(\cdot, w\odot \mathbf{1}_{\iota(K)}\right). \qquad (55)$$

Proof Let $S \in 2^r$. Observe that $f = \sum_{s=1}^r \mathbb{1}[B_s = 1]\mathbf{1}_{I_s}$ and note in particular that

$$g := \mathbf{1}_{\iota(S)}\odot f = \Big(\sum_{s\in S} + \sum_{s\in S^c}\Big)\mathbb{1}[B_s = 1]\mathbf{1}_{\iota(S)\cap I_s} = \sum_{s\in S}\mathbb{1}[B_s = 1]\mathbf{1}_{I_s}. \qquad (56)$$

Hence, g depends only on $(B_s)_{s\in S}$ and is thus moreover independent of $(B_t)_{t\in S^c}$ by assumption. Consequently $\Psi(\cdot, w\odot g)$ also depends only on $(B_s)_{s\in S}$ and is also independent of $(B_t)_{t\in S^c}$. The result then follows.

To see this in detail, suppose that $S = \{s_1, \dots, s_m\}$ and $S^c = \{t_1, \dots, t_{r-m}\}$ say. Use the law of the unconscious statistician together with (i) independence to conclude that

$$\begin{aligned}\mathbb{E}[\Psi(\cdot, w \odot g)] &\overset{(56)}{=} \sum_{b_1=0}^{1} \cdots \sum_{b_r=0}^{1} \Psi(\cdot, w \odot \sum_{s\in S} \mathbb{1}[b_s = 1]\mathbf{1}_{I_s})\mathbb{P}[B_1 = b_1, \dots, B_r = b_r] \\ &\overset{(i)}{=} \sum_{b_{s_1}=0}^{1} \cdots \sum_{b_{s_m}=0}^{1} \Psi(\cdot, w \odot \sum_{i=1}^{m} \mathbb{1}[b_{s_i} = 1]\mathbf{1}_{I_{s_i}})\mathbb{P}[\cap_{i=1}^{m}\{B_{s_i} = b_{s_i}\}] \\ &\quad \times \underbrace{\sum_{b_{t_1}=0}^{1} \cdots \sum_{b_{t_{r-m}}=0}^{1} \mathbb{P}[\cap_{j=1}^{r-m}\{B_{t_j} = b_{t_j}\}]}_{=1 \text{ as an axiom of the pdf}}. \end{aligned} \tag{57}$$

Substitute

$$\mathbb{P}[\cap_{i=1}^{m}\{B_{s_i} = b_{s_i}\}] \overset{(i)}{=} \prod_{i=1}^{m} \mathbb{P}[B_{s_i} = b_{s_i}] = \prod_{i=1}^{m} q_{s_i}^{b_{s_i}}(1 - q_{s_i})^{1-b_{s_i}} \tag{58}$$

and then apply the change of variables $K(b_{s_1}, \dots, b_{s_m}) = \cup_{i=1}^{m}\{s_i : b_{s_i} = 1\}$ to identify the right-hand side of (55). ■

We can now prove Proposition 9:

Proof (of Proposition 9) In the same notation as in Lemma 20 and Lemma 21, we use that $q_s = 1 - p_s$ is the success probability. Then, we can write

$$\begin{aligned}&\sum_{V\in 2^r} \prod_{i\in V}\Big(\frac{1}{q_i}\Big) \prod_{i\in\overline{1,r}\setminus V}\Big(1 - \frac{1}{q_i}\Big)\mathbb{E}(\Psi(\cdot, (w \odot \mathbf{1}_{\iota(V)}) \odot f) \\ &\overset{\text{(Lemma 21)}}{=} \sum_{V\in 2^r} \prod_{i\in V}\Big(\frac{1}{q_i}\Big) \prod_{i\in\overline{1,r}\setminus V}\Big(1 - \frac{1}{q_i}\Big) \sum_{K:K\subseteq V\in 2^r} \prod_{i\in K} q_i \prod_{i\in V\setminus K}(1 - q_i)\Psi(\cdot, w \odot \mathbf{1}_{\iota(K)}) \\ &= \sum_{V\in 2^r} \sum_{K:K\subseteq V\in 2^r} \prod_{i\in V}\Big(\frac{1}{q_i}\Big) \prod_{i\in\overline{1,r}\setminus V}\Big(1 - \frac{1}{q_i}\Big) \prod_{i\in K} q_i \prod_{i\in V\setminus K}(1 - q_i)\Psi(\cdot, w \odot \mathbf{1}_{\iota(K)}) \\ &= \sum_{K\in 2^r} \Psi(\cdot, w \odot \mathbf{1}_{\iota(K)}) \sum_{V:K\subseteq V\in 2^r} \Big(\frac{1}{q_i}\Big) \prod_{i\in\overline{1,r}\setminus V}\Big(1 - \frac{1}{q_i}\Big) \prod_{i\in K} q_i \prod_{i\in V\setminus K}(1 - q_i) \\ &= \sum_{K\in 2^r} \Psi(\cdot, w \odot \mathbf{1}_{\iota(K)})\mu_K \\ &\overset{\text{(Lemma 20)}}{=} \Psi(\cdot, w \odot \mathbf{1}_{\iota(\overline{1,r})}) \\ &= \Psi(\cdot, w). \end{aligned} \tag{59}$$

Note finally that we obtain Proposition 9 after substituting $q_s = 1 - p_s$. ■

Appendix B. Proofs of Section 4

In this appendix we prove the results of Section 4. In particular we prove Lemmas 13, 14 and 18 as well as Theorem 17.

B.1 Proof of Lemma 13

Let Γ be a dropout tree. Let ℓ be a leaf of Γ at level $k > 1$. Let μ be the distribution of a random matrix $F \in \{0,1\}^{d_k \times d_{k-1}}$ that satisfies for all r, c, $\mathbb{P}[F_{rc} = 1] \geq \beta > 0$. Assume that Γ satisfies $\mathsf{ApProp}_\Gamma(\delta, \epsilon)$, i.e.,

$$\mathbb{P}\Big[\sup_{x \in \overline{B(0,R)}} \sup_{\tilde{x} \in \overline{B(\mathsf{In}_\Gamma(x),\delta)}} \big|\Phi_\Gamma^{v_0}((\tilde{x}^\ell)_\ell) - \Phi_{\Gamma_{\text{det}}}^{v_0}(\mathsf{In}_\Gamma(x))\big| > \frac{\epsilon}{2}\Big] < \Big(\frac{\epsilon}{4R_L}\Big)^q.$$

Define $\kappa > 0$ by

$$\kappa := \Big(\frac{\epsilon}{4R_L}\Big)^q - \mathbb{P}\Big[\sup_{x \in \overline{B(0,R)}} \sup_{\tilde{x} \in \overline{B(\mathsf{In}_\Gamma(x),\delta)}} \big|\Phi_\Gamma^{v_0}(x) - \Phi_{\Gamma_{\text{det}}}^{v_0}(\tilde{x})\big| > \frac{\epsilon}{2}\Big].$$

By Lemma 14, there exists an $\eta > 0$ and an $N_0 \in \mathbb{N}$ such that for all $N \geq N_0$, if F^i are independent, identically distributed filter matrices distributed according to μ, and if V is given by (25), then

$$\mathbb{P}\Big[\sup_{z \in \overline{B(0,R_k)}} \sup_{(\tilde{z})^i \in B(x,\eta)^N} \Big|\sigma_k\big(\frac{1}{N}\sum_{i=1}^N (V \odot F^i)\tilde{z}^i + b^{(k)}\big) - \sigma_k\big(W^{(k)}z + b^{(k)}\big)\Big| > \delta\Big] < \kappa.$$

Now let Γ' be a μ-input-copy of Γ at ℓ of size $N \geq N_0$. Choose $\delta' := \min(\delta, \eta)$.

Consider the event $\mathcal{A}$ that

$$\sup_{x \in \overline{B(0,R)}} \sup_{\tilde{x} \in \overline{B(\mathsf{In}_\Gamma(x),\delta)}} \big|\Phi_\Gamma^{v_0}((\tilde{x}^\ell)_\ell) - \Phi_{\Gamma_{\text{det}}}^{v_0}(\mathsf{In}_\Gamma(x))\big| > \frac{\epsilon}{2},$$

which (informally) means that the tree Γ provides a bad approximation. Consider also the event $\mathcal{B}$ that

$$\sup_{z \in \overline{B(0,R_{k-1})}} \sup_{(\tilde{z}^m) \in B(z,\eta)^N} \Big|\sigma_k\big(\sum_{e \in \mathsf{into}(\ell)} (V^e \odot F^e)\tilde{z}^m + b^e\big) - \sigma_k\big(W^e z + b^e\big)\Big| > \delta.$$

Here, $\mathsf{into}(\ell)$ refers to the dropout-tree Γ'. This event (informally) means that the added part provides a bad approximation. Note that

$$\mathbb{P}[\mathcal{A} \cup \mathcal{B}] \leq \mathbb{P}[\mathcal{A}] + \mathbb{P}[\mathcal{B}] < \mathbb{P}[\mathcal{A}] + \kappa = \Big(\frac{\epsilon}{4R_L}\Big)^q.$$

Next, let us show that on $(\mathcal{A} \cup \mathcal{B})^c$ one has

$$\sup_{x \in \overline{B(0,R)}} \sup_{\tilde{x} \in \overline{B(\mathsf{In}_{\Gamma'}(x),\delta')}} \big|\Phi_{\Gamma'}^{v_0}((\tilde{x}^\ell)_\ell) - \Phi_{\Gamma_{\text{det}}}^{v_0}(\mathsf{In}_{\Gamma'}(x))\big| \leq \frac{\epsilon}{2}. \tag{60}$$

To do this, suppose that $(\mathcal{A} \cup \mathcal{B})^c$ holds and $x \in \overline{B(0,R)}$. For every leaf $m \in \mathsf{children}(\ell)$ in Γ' define $z := \mathsf{In}_{\Gamma'}^m(x)$ (recall the definition from (27)) and note that $z \in \overline{B(0,R_{k-1})}$. Let

$\tilde{x} \in B(\mathsf{In}_{\Gamma'}(x), \delta')$. For every leaf $m \in \mathsf{children}(\ell)$ define $\tilde{z}^m := \tilde{x}^m \in B(z, \eta)$ by the choice of δ'. Since $\mathcal{B}^c$ holds,

$$\Big|\sigma_k\Big(\sum_{e\in\mathsf{into}(\ell)} (V^e \odot F^e)\tilde{z}^i + b^e\Big) - \sigma_k(W^e z + b^e)\Big| \le \delta,$$

or, in other words,

$$\sigma_k\big(\sum_{e\in\mathsf{into}(\ell)} (V^e \odot F^e)\tilde{z}^i + b^e\big) \in \overline{B(\mathsf{In}^{\ell}_{\Gamma}(x), \delta)}.$$

Together with $\mathcal{A}^c$ this implies (60).

Finally, by the law of total probability, we estimate

$$\begin{aligned}
&\mathbb{P}\Big[\sup_{x\in\overline{B(0,R)}} \sup_{\tilde{x}\in\overline{B(\mathsf{In}_{\Gamma'}(x),\delta')}} \big|\Phi^{v_0}_{\Gamma'}((\tilde{x}^\ell)_\ell) - \Phi^{v_0}_{\Gamma_{\det}}(\mathsf{In}_{\Gamma'}(x))\big| > \frac{\epsilon}{2}\Big] \\
&= \mathbb{P}\Big[\sup_{x\in\overline{B(0,R)}} \sup_{\tilde{x}\in\overline{B(\mathsf{In}_{\Gamma'}(x),\delta')}} \big|\Phi^{v_0}_{\Gamma'}((\tilde{x}^\ell)_\ell) - \Phi^{v_0}_{\Gamma_{\det}}(\mathsf{In}_{\Gamma'}(x))\big| > \frac{\epsilon}{2}\Big|\mathcal{A}\cup\mathcal{B}\Big]\mathbb{P}[\mathcal{A}\cup\mathcal{B}] \\
&\quad + \mathbb{P}\Big[\sup_{x\in\overline{B(0,R)}} \sup_{\tilde{x}\in\overline{B(\mathsf{In}_{\Gamma'}(x),\delta')}} \big|\Phi^{v_0}_{\Gamma'}((\tilde{x}^\ell)_\ell) - \Phi^{v_0}_{\Gamma_{\det}}(\mathsf{In}_{\Gamma'}(x))\big| > \frac{\epsilon}{2}\Big|(\mathcal{A}\cup\mathcal{B})^c\Big]\mathbb{P}[(\mathcal{A}\cup\mathcal{B})^c] \\
&< \Big(\frac{\epsilon}{4R_L}\Big)^q + 0 = \Big(\frac{\epsilon}{4R_L}\Big)^q. \qquad (61)
\end{aligned}$$

This completes the proof of Lemma 13.

□

B.2 Proof of Lemma 14

Let $0 \le K < \infty$ and $\rho > 0$ be fixed. For every $N < \infty$, the suprema in (30) over $x, (\tilde{x}^i)$ are in fact attained—say at $X_*, (\tilde{X}^i_*)$—because σ is continuous and the optimization domain is closed and bounded. We need to now be careful because $X_*, (\tilde{X}^i_*)$ depend on the collection $\{F^i\}_{i\in\overline{1,N}}$.

Recall that the continuity of σ implies that σ is also uniformly continuous on each compact set, i.e., for every $\zeta > 0$ there exists an $\eta_\zeta > 0$ such that for all x, y from this compact set

$$|x - y| < \eta_\zeta \Rightarrow |\sigma(y) - \sigma(x)| < \zeta. \qquad (62)$$

Define $\bar{X}_* := (1/N)\sum_{i=1}^N \tilde{X}^i_*$. Then uniform continuity of σ implies that

$$\big|\sigma\big(W\bar{X}_* + b\big) - \sigma(Wx + b)\big| \le \sup_{x\in\overline{B(0,K)}} \sup_{y\in\overline{B(x,\delta)}} \big|\sigma\big(Wy + b\big) - \sigma(Wx + b)\big| =: \gamma_\delta < \infty. \qquad (63)$$

Moreover, γ_δ is independent of N. Remark also that

$$\sup_{f\in\{0,1\}^{m\times n}} \sup_{y\in\overline{B(0,\delta)}} \frac{1}{\mathbb{E}[F]}\big|\big(W\odot f - W\odot\mathbb{E}[F]\big)y\big| =: c_\delta < \infty \qquad (64)$$

where f stands for all possible deterministic realizations of the filters F. Again, c_δ is independent of N. Finally, by construction there exists a compact set $\mathcal{C} \subset \mathbb{R}^m$ such that the points

$$\frac{1}{N}\sum_{i=1}^{N}(V\odot F^i)\tilde{X}_*^i+b,\quad W\bar{X}_*+b \tag{65}$$

lie in $\mathcal{C}$ with probability one.

First, fix $\zeta=\rho/2$. From uniform continuity of σ on the compact set $\mathcal{C}$ there exists $\eta_\zeta>0$ such that (62) holds for all $x,y\in\mathcal{C}$. Second, observe that $\gamma_\delta\to 0$ and $c_\delta\to 0$ as $\delta\to 0$. Hence we can choose δ and fix it such that

$$0<\zeta<\rho-\gamma_\delta\quad\text{and}\quad c_\delta<\eta_\zeta. \tag{66}$$

Combining (63) with the triangle inequality and using (66), we arrive at

$$\text{LHS (30)}\le\mathbb{P}\Big[\underbrace{\Big|\sigma\big(\frac{1}{N}\sum_{i=1}^{N}(V\odot F^i)\tilde{X}_*^i+b\big)-\sigma\big(W\bar{X}_*+b\big)\Big|}_{=:Z}>\rho-\gamma_\delta.\Big], \tag{67}$$

Consider now the event

$$\mathcal{E}=\Big\{\Big|\frac{1}{N}\sum_{i=1}^{N}(V\odot F^i)\tilde{X}_*^i-W\bar{X}_*\Big|<\eta_\zeta\Big\}. \tag{68}$$

Then by the law of total probability and uniform continuity,

$$\begin{aligned}\mathbb{P}[Z>\rho-\gamma_\delta]&=\mathbb{P}[Z>\rho-\gamma_\delta|\mathcal{E}]\mathbb{P}[\mathcal{E}]+\mathbb{P}[Z>\rho-\gamma_\delta|\mathcal{E}^{\mathrm{c}}]\mathbb{P}[\mathcal{E}^{\mathrm{c}}]\\&\le\mathbb{1}[\zeta>\rho-\gamma_\delta]\mathbb{P}[\mathcal{E}]+\mathbb{P}[\mathcal{E}^{\mathrm{c}}]\overset{(66)}{=}\mathbb{P}[\mathcal{E}^{\mathrm{c}}].\end{aligned} \tag{69}$$

We proceed by bounding $\mathbb{P}[\mathcal{E}^{\mathrm{c}}]$. Use the triangle inequality twice to establish that for any $x\in\overline{B(0,K)}$,

$$\begin{aligned}&\Big|\frac{1}{N}\sum_{i=1}^{N}(V\odot F^i)\tilde{X}_*^i-W\bar{X}_*\Big|\\&=\Big|\frac{1}{N\mathbb{E}[F]}\sum_{i=1}^{N}\big((W\odot F^i)-W\odot\mathbb{E}[F]\big)\big(x+(\tilde{X}_*^i-x)\big)\Big|\\&\le\Big|\frac{1}{N\mathbb{E}[F]}\sum_{i=1}^{N}\big((W\odot F^i)-W\odot\mathbb{E}[F]\big)x\Big|+\Big|\frac{1}{N\mathbb{E}[F]}\sum_{i=1}^{N}\big((W\odot F^i)-W\odot\mathbb{E}[F]\big)(\tilde{X}_*^i-x)\Big|\\&\overset{(64)}{\le}\Big|\frac{1}{N\mathbb{E}[F]}\sum_{i=1}^{N}\big((W\odot F^i)-W\odot\mathbb{E}[F]\big)x\Big|+c_\delta.\end{aligned} \tag{70}$$

Note now additionally that by Khinchin's weak law of large numbers,

$$\frac{1}{N}\sum_{i=1}^{N}W\odot F^i\overset{\mathbb{P}}{\to}W\odot\mathbb{E}[F]\quad\text{as}\quad N\to\infty. \tag{71}$$

Therefore, using (66), we get as $N \to \infty$

$$\mathbb{P}[\mathcal{E}^{\mathrm{c}}] \overset{(70)}{\leq} \mathbb{P}\Big[\Big|\frac{1}{N\mathbb{E}[F]}\sum_{i=1}^{N}\big((W \odot F^i) - W \odot \mathbb{E}[F]\big)x\Big| \geq \eta_\zeta - c_\delta\Big] \overset{(71)}{\to} 0. \tag{72}$$

Bounding (69) by (72) completes the proof. □

B.3 Proof of Theorem 17

We start by showing that for α small enough and for N large enough,

$$\mathbb{P}\Big[\sup_{x\in\overline{B(0,R)}}\big|\mathsf{NN}_{\Gamma,\Xi}(x) - \Psi(x,w)\big| > \frac{\epsilon}{2}\Big] < \Big(\frac{\epsilon}{4R_L}\Big)^q. \tag{73}$$

Afterwards, we deduce the three assertions (41)–(43) from (73).

Proof of (73). Recall that the assumption $\mathsf{ApProp}_\Gamma(\delta,\epsilon)$ means that

$$\mathbb{P}\Big[\sup_{x\in\overline{B(0,R)}}\ \sup_{\tilde{x}\in\overline{B(\mathsf{In}_\Gamma(x),\delta)}}\big|\Phi^{v_0}_{\Gamma,\Xi}(\tilde{x}) - \Phi^{v_0}_{\Gamma_{\mathrm{det}},\Xi}(\mathsf{In}_\Gamma(x))\big| > \frac{\epsilon}{2}\Big] < \Big(\frac{\epsilon}{4R_L}\Big)^q.$$

Define therefore $\kappa > 0$ by

$$\kappa := \frac{1}{\#\mathsf{leaves}(\Gamma)}\bigg(\Big(\frac{\epsilon}{4R_L}\Big)^q - \mathbb{P}\Big[\sup_{x\in\overline{B(0,R)}}\ \sup_{\tilde{x}\in\overline{B(\mathsf{In}_\Gamma(x),\delta)}}\big|\Phi^{v_0}_{\Gamma,\Xi}(\tilde{x}) - \Phi^{v_0}_{\Gamma_{\mathrm{det}},\Xi}(\mathsf{In}_\Gamma(x))\big| > \frac{\epsilon}{2}\Big]\bigg). \tag{74}$$

Observe now that the function Ψ_1 is continuous, and the function $\Phi^{v_0}_{\Gamma_{\mathrm{det}}}$ is continuous on $(\mathbb{R}^{d_1})^{\mathsf{leaves}(\Gamma)}$. Since this implies uniform continuity on compact sets (see (62)), there exists a $\zeta > 0$ such that whenever a function $g : \overline{B(0,R)} \to \mathsf{Inp}_\Gamma$ satisfies

$$\sup_{x\in\overline{B(0,R)}}\ \sup_{\ell\in\mathsf{leaves}(\Gamma)}\ \big|g^\ell(x) - \Psi_1(x;(W^{(1)},b^{(1)}))\big| < \zeta,$$

then we also have

$$\sup_{x\in\overline{B(0,R)}}\Big|\Phi^{v_0}_{\Gamma_{\mathrm{det}}}(g(x)) - \Psi(x,w)\Big| < \epsilon. \tag{75}$$

Now choose

$$\rho := \min(\delta/2,\kappa,\zeta) \qquad \text{and} \qquad K := R,$$

which we use as parameters for Lemma 18. This choice ensures that for α small enough and N large enough, for all leaves ℓ of Γ, by inequality (44)

$$\mathbb{P}\Big[\sup_{x\in\overline{B(0,R)}}\big|\Xi^\ell(x) - \Psi_1(x;(W^{(1)},b^{(1)}))\big| > \delta/2\Big] < \kappa \tag{76}$$

and by inequality (45)

$$\sup_{x\in\overline{B(0,R)}}\big|\Xi^{\ell,\mathsf{avg-filt}}(x) - \Psi_1(x;(W^{(1)},b^{(1)}))\big| < \zeta. \tag{77}$$

Consider now the event $\mathcal{A}$ that there exists a leaf ℓ of Γ such that

$$\sup_{x\in\overline{B(0,R)}} |\Xi^{\ell}(x) - \Psi_1(x; (W^{(1)}, b^{(1)}))| > \delta/2.$$

From the law of total probability, it follows that

$$\begin{aligned}
&\mathbb{P}\Big[\sup_{x\in\overline{B(0,R)}} \big|\mathsf{NN}_{\Gamma,\Xi}(x) - \Psi(x,w)\big| > \frac{\epsilon}{2}\Big] \\
&= \mathbb{P}\Big[\sup_{x\in\overline{B(0,R)}} \big|\mathsf{NN}_{\Gamma,\Xi}(x) - \Psi(x,w)\big| > \frac{\epsilon}{2}\Big|\mathcal{A}\Big]\mathbb{P}[\mathcal{A}] \\
&\quad + \mathbb{P}\Big[\sup_{x\in\overline{B(0,R)}} \big|\mathsf{NN}_{\Gamma,\Xi}(x) - \Psi(x,w)\big| > \frac{\epsilon}{2}\Big|\mathcal{A}^c\Big]\mathbb{P}[\mathcal{A}^c]. \qquad (78)
\end{aligned}$$

Observe that

$$\mathbb{P}\Big[\sup_{x\in\overline{B(0,R)}} \big|\mathsf{NN}_{\Gamma,\Xi}(x) - \Psi(x,w)\big| > \frac{\epsilon}{2}\Big|\mathcal{A}\Big] \leq 1, \qquad (79)$$

and by (i) Boole's inequality

$$\begin{aligned}
\mathbb{P}[\mathcal{A}] &= \mathbb{P}\Big[\cup_{\ell\in\mathsf{leaves}(\Gamma)}\big\{\sup_{x\in\overline{B(0,R)}} |\Xi^{\ell}(x) - \Psi_1(x; (W^{(1)}, b^{(1)}))| > \delta/2\big\}\Big] \\
&\overset{\text{(i)}}{\leq} \sum_{\ell\in\mathsf{leaves}(\Gamma)} \mathbb{P}\Big[\sup_{x\in\overline{B(0,R)}} |\Xi^{\ell}(x) - \Psi_1(x; (W^{(1)}, b^{(1)}))| > \delta/2\Big] \overset{(76)}{\leq} \kappa \cdot \#\mathsf{leaves}(\Gamma). \qquad (80)
\end{aligned}$$

Furthermore,

$$\begin{aligned}
&\mathbb{P}\Big[\sup_{x\in\overline{B(0,R)}} \big|\mathsf{NN}_{\Gamma,\Xi}(x) - \Psi(x,w)\big| > \frac{\epsilon}{2}\Big|\mathcal{A}^c\Big] \\
&\overset{(75)}{\leq} \mathbb{P}\Big[\sup_{x\in\overline{B(0,R)}} \sup_{\tilde{x}\in\overline{B(\mathsf{In}_\Gamma(x),\delta)}} \big|\Phi^{v_0}_{\Gamma,\Xi}(\tilde{x}) - \Phi^{v_0}_{\Gamma_{\mathrm{det}},\Xi}(\mathsf{In}_\Gamma(x))\big| > \frac{\epsilon}{2}\Big]. \qquad (81)
\end{aligned}$$

By bounding (78) using (79)–(81) and $\mathbb{P}[\mathcal{A}^c] \leq 1$, we find that

$$\begin{aligned}
&\mathbb{P}\Big[\sup_{x\in\overline{B(0,R)}} \big|\mathsf{NN}_{\Gamma,\Xi}(x) - \Psi(x,w)\big| > \frac{\epsilon}{2}\Big] \\
&< \kappa \cdot \#\mathsf{leaves}(\Gamma) + \mathbb{P}\Big[\sup_{x\in\overline{B(0,R)}} \sup_{\tilde{x}\in\overline{B(\mathsf{In}_\Gamma(x),\delta)}} \big|\Phi^{v_0}_{\Gamma,\Xi}(\tilde{x}) - \Phi^{v_0}_{\Gamma_{\mathrm{det}},\Xi}(\mathsf{In}_\Gamma(x))\big| > \frac{\epsilon}{2}\Big] \cdot 1 \\
&\overset{(74)}{=} \Big(\frac{\epsilon}{4R_L}\Big)^q \qquad (82)
\end{aligned}$$

This shows (73).

Next, we prove that (41)–(43) follow from (73).

Proof of (41). This inequality follows from (73) since $R_L \geq 1$ by construction and $q \geq 1$ by assumption.

Proof of (43). This inequality is a direct consequence of inequality (75) by choosing $g^\ell := \Xi^{\ell,\mathsf{avg-filt}}$ and using (77).

Proof of (42). We will prove that by the definition of R_j in (26), for all $x \in \overline{B(0,R)}$

$$\big|\mathsf{NN}_{\Gamma,\Xi}(x)\big|^q < R_L^q \quad \text{w.p. one,} \quad \text{and} \quad \big|\Psi(x,w)\big|^q < R_L^q. \tag{83}$$

Namely, if (83) holds true, then (42) follows.

To see the implication, consider the event $\mathcal{D}$ for which

$$\sup_{x\in\overline{B(0,R)}} \big|\mathsf{NN}_{\Gamma,\Xi}(x) - \Psi(x,w)\big| > \frac{\epsilon}{2}, \tag{84}$$

and apply the law of total expectation:

$$\begin{aligned}
&\mathbb{E}\Big[\sup_{x\in\overline{B(0,R)}} \big|\mathsf{NN}_{\Gamma,\Xi}(x) - \Psi(x,w)\big|^q\Big] \qquad (85)\\
&= \mathbb{E}\Big[\sup_{x\in\overline{B(0,R)}} \big|\mathsf{NN}_{\Gamma,\Xi}(x) - \Psi(x,w)\big|^q \Big| \mathcal{D}\Big]\mathbb{P}[\mathcal{D}] + \mathbb{E}\Big[\sup_{x\in\overline{B(0,R)}} \big|\mathsf{NN}_{\Gamma,\Xi}(x) - \Psi(x,w)\big|^q \Big| \mathcal{D}^c\Big]\mathbb{P}[\mathcal{D}^c].
\end{aligned}$$

By the triangle inequality,

$$\mathbb{E}\Big[\sup_{x\in\overline{B(0,R)}} \big|\mathsf{NN}_{\Gamma,\Xi}(x) - \Psi(x,w)\big|^q \Big| \mathcal{D}\Big] \overset{(83)}{\leq} (2R_L)^q. \tag{86}$$

On the other hand,

$$\mathbb{E}\Big[\sup_{x\in\overline{B(0,R)}} \big|\mathsf{NN}_{\Gamma,\Xi}(x) - \Psi(x,w)\big|^q \Big| \mathcal{D}^c\Big] \overset{(84)}{\leq} \Big(\frac{\epsilon}{2}\Big)^q. \tag{87}$$

Bound now (85) using (73), (86), (87), and the elementary bound $\mathbb{P}[\mathcal{D}^c] \leq 1$ to obtain

$$\mathbb{E}\Big[\sup_{x\in\overline{B(0,R)}} \big|\mathsf{NN}_{\Gamma,\Xi}(x) - \Psi(x,w)\big|^q\Big] < (2R_L)^q\Big(\frac{\epsilon}{4R_L}\Big)^q + \Big(\frac{\epsilon}{2}\Big)^q \leq \epsilon^q.$$

That would prove (42). What remains is to prove (83).

Proof of (83). Observe immediately that the right inequality in (83) follows immediately as $0 < \beta < 1$ and $Q > 1$ (recall the definition of Ψ in (8)). Next, we will prove the left inequality in (83) by mathematical induction (recall the recursion in (34) and (35) that defines $\mathsf{NN}_{\Gamma,\Xi}$).

Base case. Recall from (34) and (35) that the induction starts with the functions

$$\Xi^\ell(x) := \sigma_1\Big(\frac{1}{N}\sum_{i=1}^{2N}(-1)^i(V^\ell \odot F^{\ell,i})\sigma_0\big((-1)^i\alpha(I \odot G^{\ell,i})x\big) + b^\ell\Big) \tag{88}$$

where element-wise

$$V_{rc}^\ell = \frac{W_{rc}^{(1)}}{\alpha\big(\sigma_- + \sigma_+\big)\mathbb{E}[F_{rc}^\ell]\mathbb{E}[G_{cc}^\ell]}. \tag{89}$$

We are now going to prove that for every $x \in \overline{B(0,R)}$, the point

$$\frac{1}{N}\sum_{i=1}^{2N}(-1)^i(V^\ell \odot F^{\ell,i})\sigma_0\big((-1)^i\alpha(I \odot G^{\ell,i})x\big) \in \overline{B\big(0,\beta^{-1}\|W^{(1)}\|_{\mathrm{HS}}R_0\big)} \quad \text{w.p. one.} \tag{90}$$

In particular, by the definition of R_1 in (26) (which implicitly deals with the bias b^ℓ), this implies that for all $x \in \overline{B(0,R)}$,

$$\big|\Xi^\ell(x)\big| \le R_1 - 1.$$

Start by noting that there exists an $\alpha_0 > 0$ such that for all $0 < \alpha \le \alpha_0$ and all $\xi \in [-R,R] \subset \mathbb{R}$ we have

$$|\sigma_0(\alpha\xi)| \le 2(|\sigma_-| + |\sigma_+|)\alpha|\xi|. \tag{91}$$

It follows from the bound (91) that for all $x \in \overline{B(0,R)}$ and all $i \in \overline{1,2N}$,

$$\Big|\frac{1}{\alpha(\sigma_- + \sigma_+)\mathbb{E}[G^\ell_{cc}]}\sigma_0\big(\alpha(I \odot G^{\ell,i})x\big)_c\Big| < 2\frac{|\sigma_-| + |\sigma_+|}{|\sigma_- + \sigma_+|}\frac{1}{\beta}|x_c| \quad \text{w.p. one.}$$

Since we assumed in (40) that

$$4\frac{|\sigma_-| + |\sigma_+|}{|\sigma_- + \sigma_+|} < Q$$

it follows that for all $x \in \overline{B(0,R)}$ and all $i \in \overline{1,2N}$,

$$\Big|2\frac{\mathbb{E}[I \odot G^\ell]^{-1}}{\alpha(\sigma_- + \sigma_+)}\sigma_0\big(\alpha(I \odot G^{\ell,i})x\big)\Big| < \frac{Q}{\beta}R < R_0 \quad \text{w.p. one.}$$

Therefore, for every $x \in \overline{B(0,R)}$,

$$\begin{aligned}
&\Big|\frac{1}{N}\sum_{i=1}^{2N}(-1)^i(V^\ell \odot F^{\ell,i})\sigma_0\big((-1)^i\alpha(I \odot G^{\ell,i})x\big)\Big| \\
&= \Big|\frac{1}{2N}\sum_{i=1}^{2N}(-1)^i((W^{(1)} \odot F^{\ell,i}) \div \mathbb{E}[F^{\ell,i}])\frac{2\mathbb{E}[I \odot G^{\ell,i}]^{-1}}{\alpha(\sigma_- + \sigma_+)}\sigma_0\big((-1)^i\alpha(I \odot G^{\ell,i})x\big)\Big| \\
&\le \frac{1}{2N}\sum_{i=1}^{2N}\|(W^{(1)} \odot F^{\ell,i}) \div \mathbb{E}[F^{\ell,i}]\|_{\mathrm{HS}}R_0 \le \frac{1}{\beta}\|W^{(1)}\|_{\mathrm{HS}}R_0 \quad \text{w.p. one.}
\end{aligned}$$

This proves (90).

Inductive step. In the definition of $\mathsf{NN}_{\Gamma,\Xi}$ we defined for v not a leaf in Γ,

$$\Phi^v_{\Gamma,\Xi} = \sigma_v\Big(\frac{1}{\#\mathsf{into}(v)}\sum_{e\in\mathsf{into}(v)}(V^e \odot F^e)\Phi^{\mathsf{source}(e)}_{\Gamma,\Xi} + b^e\Big).$$

By an inductive argument we find that for all $x \in \overline{B(0,R)}$, it holds that

$$\Big|\frac{1}{\#\mathsf{into}(v)}\sum_{e\in\mathsf{into}(v)}(V^e \odot F^e)\Phi^{\mathsf{source}(e)}_{\Gamma,\Xi}(x)\Big| \le \beta^{-1}\|W^e\|_{\mathrm{HS}}R_{\mathsf{level}(v)-1} \quad \text{w.p. one.}$$

so that by definition of $R_{\mathsf{level}(v)}$ it holds that

$$|\Phi^{v}_{\Gamma,\Xi}(x)| < R_{\mathsf{level}(v)} \quad \text{w.p. one.}$$

In particular,

$$\big|\mathsf{NN}_{\Gamma,\Xi}(x)\big|^q = \big|\Phi^{v_0}_{\Gamma,\Xi}(x)\big|^q < R_L^q \quad \text{w.p. one.}$$

This proves (83). With that, Theorem 17 is proven. □

B.4 Proof of Lemma 18

Proof of (44). Let $0 \le K < \infty$, ℓ be a leaf of Γ, and $\rho > 0$. Recall that

$$\Psi_1(x; (W^{(1)}, b^{(1)})) = \sigma_1\big(W^{(1)}x + b^{(1)}\big), \tag{92}$$

and for $x \in \overline{B(0,K)}$, define

$$Z^{\ell}_N(x) = \frac{1}{N}\sum_{i=1}^{2N}(-1)^i(V^{\ell} \odot F^{\ell,i})\sigma_0\big((-1)^i\alpha(I \odot G^{\ell,i})x + b\big) + b^{\ell} \tag{93}$$

so that $\Xi^{\ell}(x) = \sigma_1(Z^{\ell}_N(x))$. Note that the weights W are fixed and therefore uniformly bounded.

Continuity of σ_0 and σ_1, boundedness of F and G, positivity of $\mathbb{E}[F^{\ell}_{rc}]$ and $\mathbb{E}[G^{\ell}_{rc}]$, and compactness of the optimization domain imply that the supremum of the optimization problem is attained—say at $X_* \in \overline{B(0,K)}$. Just like in Appendix B.2, note that X_* is random and depends on the collections $\{F^{\ell,i}\}_{i\in\overline{1,2N}}$, $\{G^{\ell,i}\}_{i\in\overline{1,2N}}$. Summarizing:

$$\begin{aligned}\mathbb{P}\Big[\sup_{x\in\overline{B(0,K)}}\big|\sigma_1(Z^{\ell}_N(x)) - \sigma_1(W^{(1)}x + b^{(1)})\big| > \rho\Big]\\ = \mathbb{P}\Big[\big|\sigma_1(Z^{\ell}_N(X_*)) - \sigma_1(W^{(1)}X_* + b^{(1)})\big| > \rho\Big].\end{aligned} \tag{94}$$

Note that here we slightly abuse the notation by using $|\cdot|$ sign not only for absolute value of numbers, but also, as in the last formula, for the Euclidean norm of the vector.

By construction, there exists a compact set $\mathcal{C}$ so that the points

$$Z^{\ell}_N(X_*), \quad W^{(1)}X_* + b^{(1)} \tag{95}$$

lie in $\mathcal{C}$ with probability one. The uniform continuity of σ_1 on $\mathcal{C}$ implies that for each $\zeta > 0$ there exists $\eta_\zeta > 0$ such that (62) holds for σ_1 and all $x, y \in \mathcal{C}$. Fix $\zeta = \rho$ and introduce the event

$$\mathcal{D}(X_*) = \big\{\|Z^{\ell}_N(X_*) - (W^{(1)}X_* + b^{(1)})\|_2 < \eta_\rho\big\}. \tag{96}$$

By the law of total probability

$$\begin{aligned}&\mathbb{P}\Big[|\sigma_1(Z^{\ell}_N(X_*)) - \sigma_1(W^{(1)}X_* + b^{(1)})| > \rho\Big]\\ &= \mathbb{P}\Big[|\sigma_1(Z^{\ell}_N(X_*)) - \sigma_1(W^{(1)}X_* + b^{(1)})| > \rho \,|\, \mathcal{D}(X_*)\Big]\mathbb{P}\Big[\mathcal{D}(X_*)\Big]\\ &\quad + \mathbb{P}\Big[|\sigma_1(Z^{\ell}_N(X_*)) - \sigma_1(W^{(1)}X_* + b^{(1)})| > \rho \,|\, \mathcal{D}^{\mathrm{c}}(X_*)\Big]\mathbb{P}\Big[\mathcal{D}^{\mathrm{c}}(X_*)\Big] \le \mathbb{P}\Big[\mathcal{D}^{\mathrm{c}}(X_*)\Big].\end{aligned} \tag{97}$$

We will next prove that for all $x \in \overline{B(0,K)}$,

$$\mathbb{P}\Big[\mathcal{D}^{\mathrm{c}}(x)\Big] \to 0 \tag{98}$$

as $\alpha \downarrow 0$ and $N \to \infty$. Together with (97), this implies the result.

Let $x \in \overline{B(0,K)}$. Component-wise,

$$\begin{aligned}&\big(Z_N^\ell(x) - (W^{(1)}x + b^{(1)})\big)_r \qquad (99)\\ &= \Big(\sum_{i=1}^{2N} \frac{(-1)^i}{N} (V^\ell \odot F^{\ell,i}) \sigma_0\big((-1)^i \alpha (I \odot G^{\ell,i})x\big) + b^\ell - (W^{(1)}x + b^{(1)}) \Big)_r \\ &= \sum_{i=1}^{2N} \sum_{c=1}^{d_0} \frac{(-1)^i}{N} V_{rc}^\ell F_{rc}^{\ell,i} \sigma_0\big((-1)^i \alpha (I \odot G^{\ell,i})x\big)_c + b_r^{(1)} - \sum_c W_{rc}^{(1)} x_c + b_r^{(1)}.\end{aligned}$$

Substituting (39) into (99), using the triangle inequality, and rearranging terms, we find that

$$\begin{aligned}&\big|\big(Z_N^\ell(x) - (W^{(1)}x + b^{(1)})\big)_r\big| \\ &\leq \sum_{c=1}^{d_0} \Big| \frac{W_{rc}^{(1)}}{\frac{1}{2}(\sigma_- + \sigma_+)\mathbb{E}[G_{cc}^\ell]} \Big(\tfrac{1}{2N} \sum_{i=1}^{2N} \frac{F_{rc}^{\ell i}}{\mathbb{E}[F_{rc}^\ell]} (-1)^i \sigma_0\big((-1)^i \alpha (I \odot G^{\ell,i})x\big)_c \Big) - W_{rc}^{(1)} x_c \Big|. \qquad (100)\end{aligned}$$

Note that the assumptions of the lemma imply that

$$\Big| \frac{W_{rc}^{(1)}}{\frac{1}{2}(\sigma_- + \sigma_+)\mathbb{E}[G_{cc}^\ell]} \Big| \leq C_{w,G,\sigma} < +\infty.$$

We focus now on the term within brackets in (100). Let $\delta_1 > 0$ and consider the event

$$\mathcal{E}_{F,N}(\delta_1) = \Big\{ \Big| \tfrac{1}{2N} \sum_{i=1}^{2N} \frac{F^{\ell i}}{\mathbb{E}[F_{rc}^\ell]} - 1 \Big| < \delta_1 \Big\}. \tag{101}$$

There exists $C_1 > 0$ such that, conditional on $\mathcal{E}_{F,N}(\delta_1)$,

$$\begin{aligned}&\Big| \tfrac{1}{2N} \sum_{i=1}^{2N} \frac{F^{\ell i}}{\mathbb{E}[F_{rc}^\ell]} (-1)^i \sigma_0\big((-1)^i \alpha (I \odot G^{\ell,i})x\big)_c - \tfrac{1}{2N} \sum_{i=1}^{2N} (-1)^i \sigma_0\big((-1)^i \alpha (I \odot G^{\ell,i})x\big)_c \Big| \\ &\leq \Big| \tfrac{1}{2N} \sum_{i=1}^{2N} \Big(\frac{F^{\ell i}}{\mathbb{E}[F_{rc}^\ell]} - 1 \Big) (-1)^i \sigma_0\big((-1)^i \alpha (I \odot G^{\ell,i})x\big)_c \Big| \leq C_1 \delta_1 \qquad (102)\end{aligned}$$

since the argument of σ_0 is uniformly bounded, and σ_0 is continuous. Moreover, there exists $C_2 > 0$ such that conditional on $\mathcal{E}_{F,N}(\delta_1)$,

$$\begin{aligned}&\big|\big(Z_N^\ell(x) - (W^{(1)}x + b^{(1)})\big)_r\big| \qquad (103)\\ &\leq \sum_{c=1}^{d_0} \Big| \frac{W_{rc}^{(1)}}{\frac{1}{2}(\sigma_- + \sigma_+)\mathbb{E}[G_{cc}^\ell]} \Big(\tfrac{1}{2N} \sum_{i=1}^{2N} \tfrac{1}{\alpha} \Big((-1)^i \sigma_0\big((-1)^i \alpha (I \odot G^{\ell,i})x\big)_c \Big) \Big) - W_{rc}^{(1)} x_c \Big| + C_2 \delta_1.\end{aligned}$$

Note that C_1, C_2 are independent of N, δ_1.

Recall now that by (36) and (37) we can find for each $\gamma > 0$ an $\alpha > 0$ such that for all $y \in \mathbb{R}^{d_0}$, c,

$$\Big| \frac{1}{\alpha} \Big(\sigma_0 \big(\alpha y \big) \Big)_c - \sigma_{S(y_r)} y_c \Big| < \gamma. \tag{104}$$

Recall furthermore that $\max_{rc} G^{\ell}_{rc} \leq M < \infty$ with probability one by assumption. Together, this implies that we can find for each $\gamma > 0$ an $\alpha > 0$ such that for all $x \in \overline{B(0,K)}$, c,

$$\mathbb{P}\Big[\Big| \frac{1}{\alpha} \Big(\pm \sigma_0 \big(\pm \alpha I \odot G^{\ell,i} x \big) \Big)_c - \sigma_{S(\pm G^{\ell i}_{cc} x_c)} G^{\ell i}_{cc} x_c \Big| < \gamma \Big] = 1 \tag{105}$$

Fix $\gamma \in (0, \delta_1)$ and corresponding $\alpha > 0$. Then there exists a constant $C_3 > 0$, independent of δ_1, γ, N, such that conditional on $\mathcal{E}_{F,N}(\delta_1)$,

$$\begin{aligned} &\big| \big(Z^{\ell}_N(x) - (W^{(1)} x + b^{(1)}) \big)_r \big| \\ &\leq \sum_{c=1}^{d_0} \Big| \frac{W^{(1)}_{rc}}{\frac{1}{2}(\sigma_- + \sigma_+) \mathbb{E}[G^{\ell}_{cc}]} \Big(\tfrac{1}{2N} \sum_{i=1}^{2N} \sigma_{S((-1)^i G^{\ell i}_{cc} x_c)} G^{\ell i}_{cc} x_c \Big) - W^{(1)}_{rc} x_c \Big| + C_3 \delta_1 \\ &\overset{\text{(i)}}{=} \sum_{c \in \overline{1, d_0} : x_c > 0} \Big| \frac{W^{(1)}_{rc}}{\frac{1}{2}(\sigma_- + \sigma_+) \mathbb{E}[G^{\ell}_{cc}]} \Big(\tfrac{1}{2N} \sum_{i=1}^{2N} \sigma_{S((-1)^i G^{\ell i}_{cc} x_c)} G^{\ell i}_{cc} x_c \Big) - W^{(1)}_{rc} x_c \Big| + C_3 \delta_1. \end{aligned} \tag{106}$$

To conclude (i), we used the fact that $\sigma_{\pm} \cdot 0 = 0$. By assumption $G^{\ell i}_{cc} \geq 0$ with probability one, so if moreover $|x_c| > 0$, then $S((-1)^i G^{\ell i}_{cc} x_c) = S((-1)^i x_c)$ with probability one—recall its definition in (37). Thus there exists C_4 independent of δ_1, γ, N such that conditional on the event $\mathcal{E}_{F,N}(\delta_1) \cap \mathcal{E}_{G,N}(\delta_1)$,

$$\begin{aligned} &\big| Z^{\ell}_N(x) - (W^{(1)} x + b^{(1)}) \big|_r \\ &\leq \sum_{c \in \overline{1, d_0} : |x_c| > 0} \Big| \frac{W^{(1)}_{rc}}{\frac{1}{2}(\sigma_- + \sigma_+)} \frac{1}{2N} \sum_{i=1}^{2N} \sigma_{S((-1)^i x_c)} x_c - W^{(1)}_{rc} x_c \Big| + C_4 \delta_1 = C_4 \delta_1. \end{aligned} \tag{107}$$

The last equality holds because the sum is only of over c such that $|x_c| > 0$.

All that remains is to prove that

$$\mathbb{P}[\mathcal{E}_{F,N}(\delta_1) \cap \mathcal{E}_{G,N}(\delta_1)] \to 1 \quad \text{as} \quad N \to \infty. \tag{108}$$

This fact follows immediately from the independence of F, G, and a subsequent application of Khinchin's weak law of large numbers (which may be applied since F, G's expectations are bounded). Note that δ_1 is an arbitrary parameter: choosing it such that $C_4 \delta_1 < \eta_\zeta$, and then choosing N sufficiently large completes the proof of (44).

Proof of (45). The assertion (42) is proven for any (α, N)-precomposition associated with some distributions μ^{ℓ}, ν^{ℓ} with finite nonzero mean. In particular, the same argument shows that (42) holds when F^{ℓ} and G^{ℓ} are taken deterministic and equal to the expectations of the corresponding random variables (see also the discussion following (39)). This proves (45). □

References

George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

Gwendoline De Bie, Gabriel Peyré, and Marco Cuturi. Stochastic deep networks. In *International Conference on Machine Learning*, pages 1556–1565. PMLR, 2019.

Dennis Elbrächter, Dmytro Perekrestenko, Philipp Grohs, and Helmut Bölcskei. Deep neural network approximation theory. *IEEE Transactions on Information Theory, submitted Jan. 2019, revised*, June 2020. URL `http://www.nari.ee.ethz.ch/pubs/p/deep-it-2019`.

Andrew Foong, David Burt, Yingzhen Li, and Richard Turner. On the expressiveness of approximate inference in Bayesian neural networks. *Advances in Neural Information Processing Systems*, 33, 2020.

Claudio Gallicchio and Simone Scardapane. Deep randomized neural networks. In *Recent Trends in Learning From Data*, pages 43–68. Springer, 2020.

Erol Gelenbe, Zhi-Hong Mao, and Yan-Da Li. Function approximation with spiked random networks. *IEEE Transactions on Neural Networks*, 10(1):3–9, 1999a.

Erol Gelenbe, Zhi-Wong Mao, and Yan-Da Li. Approximation by random networks with bounded number of layers. In *Neural Networks for Signal Processing IX: Proceedings of the 1999 IEEE Signal Processing Society Workshop (Cat. No. 98TH8468)*, pages 166–175. IEEE, 1999b.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2):251–257, 1991.

Boris Igelnik and Yoh-Han Pao. Stochastic choice of basis functions in adaptive function approximation and the functional-link net. *IEEE Transactions on Neural Networks*, 6 (6):1320–1329, 1995.

Alex Labach, Hojjat Salehinejad, and Shahrokh Valaee. Survey of dropout methods for deep neural networks. *arXiv preprint arXiv:1904.13310*, 2019.

Moshe Leshno, Vladimir Ya Lin, Allan Pinkus, and Shimon Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural networks*, 6(6):861–867, 1993.

Yuly Makovoz. Random approximants and neural networks. *Journal of Approximation Theory*, 85(1):98–109, 1996.

Yuly Makovoz. Uniform approximation by neural networks. *Journal of Approximation Theory*, 95(2):215–228, 1998.

Hien Nguyen. A universal approximation theorem for Gaussian-gated mixture of experts models. *Available at SSRN 2946964*, 2017.

Hien D. Nguyen, Luke R. Lloyd-Jones, and Geoffrey J. McLachlan. A universal approximation theorem for mixture-of-experts models. *Neural computation*, 28(12):2585–2593, 2016.

Yoh-Han Pao, Gwang-Hoon Park, and Dejan J. Sobajic. Learning and generalization characteristics of the random vector functional-link net. *Neurocomputing*, 6(2):163–180, 1994.

Allan Pinkus. Approximation theory of the MLP model in neural networks. *Acta numerica*, 8:143–195, 1999.

Ali Rahimi and Benjamin Recht. Uniform approximation of functions with random bases. In *2008 46th Annual Allerton Conference on Communication, Control, and Computing*, pages 555–561. IEEE, 2008.

Stelios Timotheou. The random neural network: a survey. *The computer journal*, 53(3): 251–267, 2010.

Stefan Wager, Sida Wang, and Percy S Liang. Dropout training as adaptive regularization. In *Advances in Neural Information Processing Systems*, pages 351–359, 2013.

Li Wan, Matthew Zeiler, Sixin Zhang, Yann Le Cun, and Rob Fergus. Regularization of neural networks using Dropconnect. In *International Conference on Machine Learning*, pages 1058–1066, 2013.

Halbert White. An additional hidden unit test for neglected nonlinearity in multilayer feedforward networks. In *Proceedings of the international joint conference on neural networks*, volume 2, pages 451–455. Washington, DC, 1989.

Yonghua Yin. Random neural network methods and deep learning. *Probability in the Engineering and Informational Sciences*, pages 1–31, 2019.