Concentration inequalities and optimal number of layers for stochastic deep neural networks

Anonymous authors
Paper under double-blind review

Abstract

We state concentration and martingale inequalities for the output of the hidden layers of a stochastic deep neural network (SDNN), as well as for the output of the whole SDNN. These results allow us to introduce an expected classifier (EC), and to give probabilistic upper bound for the classification error of the EC. We also state the optimal number of layers for the SDNN via an optimal stopping procedure. We apply our analysis to a stochastic version of a feedforward neural network with ReLU activation function.

1 Introduction

Deep neural networks (DNNs) are used extensively in modern statistics and machine learning due to their prediction accuracy; they are applied across many different areas of artificial intelligence, computer vision, speech recognition, and natural language processing (Bahdanau et al., 2016; Hinton et al., 2012; Kalchbrenner & Blunsom, 2013; Krizhevsky et al., 2017; LeCun et al., 2015). Nonetheless, it is common knowledge that the theoretical understanding of many of their properties is still incomplete (for an account on recent results, see Zhang et al. (2018, Section 1)).

In contrast to classical neural networks – which learn mappings from a set of inputs to a set of outputs – stochastic neural networks (SNNs) learn mappings from a set of inputs to a set of probability distributions over the set of outputs. The added complexity introduced by the stochasticity exacerbates the study of their theoretical properties. Stochastic neural networks have several advantages with respect to deterministic DNNs: they have greater expressive power as they allow for multi-modal mappings, and the stochasticity can be seen as added regularization (Lee et al., 2017). In Jospin et al. (2022), the authors point out that the main goal of using SNNs is to obtain a better idea of the uncertainty associated with the underlying process.

Stochastic neural networks were introduced by Wong (1991) to generalize Hopfield networks (Hopfield, 1982). The theoretical features of stochastic deep neural networks (SDNNs) have been the object of study of many recent papers. In Merkh & Montúfar (2019), the authors study universal approximation properties of deep belief networks, a class of stochastic feedforward networks that were pivotal in the resurgence of deep learning. A key observation is that, for a finite number of inputs and outputs, the number of maps in the stochastic setting is infinite, unlike in the deterministic framework. This points to the massive increase in the approximation power of SDNNs. In De Bie et al. (2019), SDNNs are framed as learning measures (LMs); deep architectures are introduced to address issues that arise in LMs such as permutation invariance or equivariance and variation in weights. Training stochastic feedforward networks is significantly more challenging. To address this issue, Tang & Salakhutdinov (2013) proposed a stochastic feedforward network with hidden layers composed of both deterministic and stochastic variables with a novel generalized EM training procedure that allows the user to efficiently learn complicated conditional distributions.

In this work, we first investigate concentration inequalities for the hidden layers of a SDNN.

Theorem 1. *Let $(\nu^{(l)}(X))_{l=1}^{L}$ denote the sequence of outputs of the hidden layers of a generic SDNN. If*

- *either $\nu^{(l)}(X)$ is bounded*
- *or the sequence of centered outputs $(\nu^{(l)}(X) - \mathbb{E}[\nu^{(l)}(X)])_{l=1}^{L}$ is a smooth weak martingale,*

then $\nu^{(l)}(X)$ concentrates around its expected value.

We prove Theorem 1 in sections 2.1 and 2.2. In this latter we also state conditions that allow us verify the martingale hypothesis. Furthermore, we prove the following bound on the classification error of the expected classifier.

Theorem 2. *If the output $s(\nu(X))$ of the score function that we choose for our analysis is bounded, then the score concentrates around its expected value $\mathbb{E}[s(\nu(X))]$. The classification error obtained using a classifier based on $\mathbb{E}[s(\nu(X))]$ is small.*

We prove Theorem 2 in section 2.3. In addition, we provide a procedure to find the optimal number of layers that draws from Peskir & Shiryaev (2006, Section 1.2), and we apply our findings to a stochastic version of the feedforward neural network with ReLU activation studied in Zhang et al. (2018).

1.1 Previous work

Concentration inequalities for SDNNs have been studied in the context of PAC-Bayes bounds and stochastic gradient descent (SGD) solutions. In particular, Huang et al. (2020) propose the Kronecker Flow, an invertible transformation-based method that generalizes the Kronecker product to a nonlinear formulation, and uses this construction to tighten PAC-Bayesian bounds. They show that the KL divergence in the PAC-Bayes bound can be estimated with high probability (they give a Hoeffding-type concentration result), and demonstrate the generalization gap can be further reduced and explained by leveraging structure in parameter space. Zhu et al. (2022) study the concentration property of SGD solutions. They consider a very rich class of gradient noise – not imposing restrictive requirements such as boundedness or sub-Gaussianity – where only finitely-many moments are required, thus allowing heavy-tailed noises. In the present work, we focus on concentration inequalities (of the Chernoff type) for the output of the hidden layers and of the whole SDNN.

The other focus of our work is finding the number of layers of a generic SDNN that strikes a balance between computational cost and accuracy of the analysis. A similar problem was studied by Trelin & Prochazka (2019). There, the authors present Binary Stochastic Filtering (BSF), an algorithm for feature selection and neuron pruning in a special version of the classical deep neural network structure where stochastic neurons are mixed with deterministic ones. To the best of our knowledge, the present paper is the first to present an optimal stopping procedure to select the number of layers in a generic SDNN.

1.2 Contributions

The first goal of this paper is to give concentration inequalities for the outputs of the hidden layers of a generic SDNN. Although this problem has already been studied for particular types of SDNNs (Garnier & Langhendries, 2021; Merkh & Montúfar, 2019; Ost & Reynaud-Bouret, 2020), we analyze it in great generality, requiring very little mathematical structure. In Theorem 3 we show that, under a reasonable assumption, the output $\nu^{(1)}(X)$ of the first layer $\nu^{(1)}$ of SDNN ν concentrates around its expected value. In Corollary 4 we show that this is true for the outputs of all the layers, so in turn it is also true for the output of the neural network. In Corollary 6 we show how, if the neural network is a weak martingale, then under a weaker assumption than the one in Theorem 3, the output $\nu^{(l)}(X)$ of the l-th layer $\nu^{(l)}$ of SDNN ν concentrates around its expected value. In Theorem 8 we give a sufficient condition for our neural network to be a weak martingale. In Proposition 11 we show that, under a mild assumption, the classifier of our stochastic neural network concentrates around its expected value. This gives us an *expected decision boundary* (EDB). In Proposition 12 we give a probabilistic bound to the classification error of the classifier based on the EBD, that we call the *expected classifier* (EC).

We then turn our attention to the number of layers to select. When specifying the structure of a generic DNN ν, we face a trade-off between accuracy and computational efficiency. One of the main drivers of this trade-off is the number of layers: more layers may yield more accurate analysis, but they also training more computationally intensive. To solve this problem, Liu & Deng (2018) introduce a new type of DNN called a Dynamic Deep Neural Network (D^2NN) that selects a subset of neurons to execute computations.

Given an input, only a subset of D^{2}NN neurons are executed, and the particular subset is determined by the D^{2}NN itself. Sabuncu (2020) points out that the problem of selecting the correct number of layers has been studied also when stochasticity is featured in the architecture design of the neural network. For example, Huang et al. (2016) study DNNs with stochastic depth: the number of layers is chosen randomly, while in Li & Talwalkar (2020); Xie et al. (2020) the authors use a random search algorithm to optimize over the space of neural network architectures. Given a generic SDNN, we adopt an optimal stopping approach to the problem. In Theorem 14, we find the number τ_1^L of layers for the SDNN that optimizes the trade-off between accuracy and computational cost. To the best of our knowledge, this is the first time such a problem is studied via an optimal stopping procedure for generic SDNNs.

Finally, we apply our findings to a stochastic feedforward neural networks with ReLU activation. We generalize the setup in Zhang et al. (2018) – that builds a bridge between tropical geometry and deep neural networks – by allowing the parameters of the neural network to be stochastic. In particular, in every layer of the neural network we let the weight matrix and the bias vector be stochastic and possibly correlated with the ones in the previous layers. In Proposition 15 we show that the assumption of Theorem 3 is easily satisfied in this setup. In Proposition 18 we prove that the upper bounds for the number of positive and negative regions of the stochastic version of the neural network found in Zhang et al. (2018) concentrate around their expected value. Although the tropical geometry of deep neural networks has been further investigated (Alfarra et al., 2021; Maragos et al., 2021), this is the first time probabilistic results are presented within the framework introduced by Zhang et al. (2018).

The paper is organized as follows. In section 2 we provide concentration inequalities for the hidden layers of a SDNN and for its output. In section 3 we provide results for the optimal number of layers. In section 4 we apply our results to a stochastic version of the feedforward neural networks with ReLU activation presented in Zhang et al. (2018). Section 5 is a discussion. To make the paper self-contained we provide background on subGaussian random variables and norm-subGaussian random vectors in Appendix A, and on martingales and filtrations in Appendix B. We prove our results in Appendix C.

1.3 Notation

We introduce the notation for deep neural networks (DNNs) that we use throughout the paper. An L-layered DNN is a map $\nu : \mathbb{R}^d \to \mathbb{R}^p$. We denote the *width* of the l-th layer, that is, the number of nodes of the l-th layer, by n_l, $l \in \{0, \dots, L\}$, where $n_0 = d$ and $n_L = p$, the dimensions of the input and the output of the network, respectively. The output of the l-th layer is given by $\nu^{(l)} : \mathbb{R}^{n_{l-1}} \to \mathbb{R}^{n_l}$. We assume for convenience that $\nu^{(0)}(x) \equiv x$. The final output $\nu(x) \equiv \nu^{(L)}(x)$ of the neural network is fed into a *score function* $s : \mathbb{R}^p \to \mathbb{R}^m$ that is application specific. We call $\mathcal{X}$ the space of inputs, $\mathcal{Y}$ the space of responses, and $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$. We assume we collect data $(X, Y) \sim \mathscr{D}$, a distribution on $\mathcal{Z}$. In a SDNN, $(\nu^{(l)}(X))_{l=0}^L$ is a sequence of random vectors having possibly different dimensions and being possibly correlated. Notice that $\nu^{(l)}(x)$ is the realization of random vector $\nu^{(l)}(X) : \Omega \to \mathbb{R}^{n_l}$ whose elements may be correlated. Throughout this paper, we assume that $\nu^{(l)}(X)$ has a finite first moment, for all $l \in \{1, \dots, L\}$.

2 Concentration inequalities

In this section we derive concentration inequalities for the hidden layers of a generic SDNN.

2.1 Norm-subGaussian-based concentration inequalities

The results in this section rely on properties of subGaussian random variables and norm-subGaussian random vectors, see Appendix A for details. Let $\|\cdot\|_2$ denote the Euclidean norm; the following is the first result.

Theorem 3. *Suppose that $\nu^{(1)}(X)$ is bounded so that $\|\nu^{(1)}(X)\|_2 \leq \sigma_{(1)}$, for some $\sigma_{(1)} \in \mathbb{R}$. Then, for all $t \in \mathbb{R}$,*

$$P_{(X,Y)\sim\mathscr{D}}\left(\|\nu^{(1)}(X) - \mathbb{E}_{(X,Y)\sim\mathscr{D}}\left[\nu^{(1)}(X)\right]\|_2 \geq t\right) \leq 2\exp\left(-\frac{t^2}{2\sigma_{(1)}^2}\right). \tag{1}$$

Theorem 3 tells us that the output $\nu^{(1)}(X)$ of the first layer of our SDNN concentrates around its expected value $\mathbb{E}_{(X,Y)\sim\mathscr{D}}[\nu^{(1)}(X)]$. The assumption that $\nu^{(1)}(X)$ is bounded is mild: it can be interpreted as a safety check to ensure that output $\nu^{(1)}(X)$ does not take on values that are too extreme.

We can use Theorem 3 to find a concentration inequality for the second layer of our neural network. In particular, if $\|\nu^{(2)}(X)\|_2 \leq \sigma_{(2)}$, for some $\sigma_{(2)} \in \mathbb{R}$ possibly different from $\sigma_{(1)}$, we have that, for all $t \in \mathbb{R}$,

$$P_{(X,Y)\sim\mathscr{D}}\left(\|\nu^{(2)}(X) - \mathbb{E}_{(X,Y)\sim\mathscr{D}}\left[\nu^{(2)}(X)\right]\|_2 \geq t\right) \leq 2\exp\left(-\frac{t^2}{2\sigma^2_{(2)}}\right). \tag{2}$$

This tells us that the output $\nu^{(2)}(X)$ of the second layer of our SDNN concentrates around its expected value $\mathbb{E}[\nu^{(2)}(X)]$.

The following corollary tells us that the result in Theorem 3 holds for every layer of our SDNN.

Corollary 4. *Pick any $l \in \{1,\dots,L\}$. Suppose that $\nu^{(l)}(X)$ is bounded so that $\|\nu^{(l)}(X)\|_2 \leq \sigma_{(l)}$, for some $\sigma_{(l)} \in \mathbb{R}$. Then, for all $t \in \mathbb{R}$,*

$$P_{(X,Y)\sim\mathscr{D}}\left(\|\nu^{(l)}(X) - \mathbb{E}_{(X,Y)\sim\mathscr{D}}\left[\nu^{(l)}(X)\right]\|_2 \geq t\right) \leq 2\exp\left(-\frac{t^2}{2\sigma^2_{(l)}}\right). \tag{3}$$

As a result of Corollary 4, we have that the output $\nu(X) = \nu^{(L)}(X)$ of our SDNN concentrates around its expected value $\mathbb{E}_{(X,Y)\sim\mathscr{D}}[\nu(X)] = \mathbb{E}_{(X,Y)\sim\mathscr{D}}[\nu^{(L)}(X)]$.

2.2 Martingale inequalities

We assume now that the sequence $\boldsymbol{\nu} = (\nu^{(l)}(X))_{l=0}^{L}$ of outputs of the hidden layers of our SDNN is a martingale; this allows us to state some interesting properties of SDNNs. Note that if $\boldsymbol{\nu}$ is any kind of martingale (very-weak, weak, or strong, see Appendix B), then the width of the layers of the SDNN is constant throughout the neural network and equal to p. Indeed, if $n_l = n_{l-1} = p$ does not hold for all $l \in \{0,\dots,L\}$, then computing $\mathbb{E}[\nu^{(l)}(X) \mid \nu^{(0)}(X),\dots,\nu^{(l-1)}(X)]$ may not have a clear mathematical meaning. Neural networks with constant width are used extensively and a well studied subject (Telgarsky, 2016).

We now show how a martingale hypothesis allows us to weaken the already mild assumption in Theorem 3 and Corollary 4. The following theorem is an immediate result of Hayes (2005, Theorem 1.8).

Theorem 5. *If $\boldsymbol{\nu}$ is a weak martingale in $\mathbb{R}^p$ such that for every $l \in \{1,\dots,L\}$, $\|\nu^{(l)}(X) - \nu^{(l-1)}(X)\|_2 \leq M$, for some $M > 0$, then, for every $a > 0$,*

$$P_{(X,Y)\sim\mathscr{D}}\left(\|\nu^{(l)}(X)\|_2 \geq Ma\right) < 2\exp\left(\frac{1-(Ma-1)^2}{2l}\right). \tag{4}$$

The fact that we demand the distance between the output of successive layers of the neural network to be bounded can be interpreted as the mild requirement for the SDNN to be "smooth". That is, no steep jumps from the output of one layer to the output of the successive one are allowed.

Consider now the sequence $\mathbf{Y} = (Y^{(l)})_{l=0}^{L}$ such that $Y^{(l)} = \nu^{(l)}(X) - \mathbb{E}_{(X,Y)\sim\mathscr{D}}[\nu^{(l)}(X)]$, for all $l \in \{1,\dots,L\}$. Then, we have the following.

Corollary 6. *If $\mathbf{Y}$ is a weak martingale in $\mathbb{R}^p$ such that for every $l \in \{1,\dots,L\}$, $\|Y^{(l)} - Y^{(l-1)}\|_2 \leq M$, for some $M > 0$, then, for every $a > 0$,*

$$\begin{aligned} P_{(X,Y)\sim\mathscr{D}}\left(\|Y^{(l)}\|_2 \geq Ma\right) &= P_{(X,Y)\sim\mathscr{D}}\left(\|\nu^{(l)}(X) - \mathbb{E}_{(X,Y)\sim\mathscr{D}}\left[\nu^{(l)}(X)\right]\|_2 \geq Ma\right) \\ &< 2\exp\left(\frac{1-(Ma-1)^2}{2l}\right). \end{aligned} \tag{5}$$

Requiring the distance between $Y^{(l)}$ and $Y^{(l-1)}$ to be bounded for all l retains the "smoothness" interpretation we have given before: we do not want jumps that are too steep between successive centered outputs of the layers. Corollary 6 tells us that $\nu^{(l)}(X)$ concentrates around $\mathbb{E}_{(X,Y)\sim\mathscr{D}}[\nu^{(l)}(X)]$, for all $l \in \{1,\dots,L\}$. In turn, this provides conditions under which the output of the neural network concentrates around its expected value.

There is one main difference between Corollary 6 and Corollary 4. In the latter we require $\nu^{(l)}(X)$ to be bounded, while in the former we only require that the Euclidean distance between $\nu^{(l)}(X) - \mathbb{E}_{(X,Y)\sim\mathscr{D}}[\nu^{(l)}(X)]$ and $\nu^{(l-1)}(X) - \mathbb{E}_{(X,Y)\sim\mathscr{D}}[\nu^{(l-1)}(X)]$ is bounded. This is a milder assumption since it governs the dynamics of the sequence of layer outputs, rather than of the outputs themselves. It comes at the cost of verifying that the sequence of outputs in each layer is a weak martingale.

We now provide two theorems that allow us to check whether sequences $\boldsymbol{\nu}$ or $\mathbf{Y}$ are weak or very-weak martingales. We first need a definition.

Definition 7. *Consider two generic p-dimensional random vectors X_1, X_2. If*

$$\mathbb{E}(\phi(X_1)) \leq \mathbb{E}(\phi(X_2))$$

for all convex functions $\phi : \mathbb{R}^p \to \mathbb{R}$, provided the expectations exist, then X_1 is smaller than X_2 in the convex order, denoted by $X_1 \leq_{cx} X_2$.

The following results come from Shaked & Shanthikumar (2007, Theorem 7.A.1).

Theorem 8. *Pick any $l \in \{1,\dots,L\}$ and any $j < l$. If $\nu^{(j)}(X) \leq_{cx} \nu^{(l)}(X)$, then $\boldsymbol{\nu}$ is a weak martingale. Similarly, if $Y^{(j)} \leq_{cx} Y^{(l)}$, then $\mathbf{Y}$ is a weak martingale.*

Theorem 9. *Pick any $l \in \{1,\dots,L\}$. If $\nu^{(l-1)}(X) \leq_{cx} \nu^{(l)}(X)$, then $\boldsymbol{\nu}$ is a very-weak martingale. Similarly, if $Y^{(l-1)} \leq_{cx} Y^{(l)}$, then $\mathbf{Y}$ is a very-weak martingale.*

To the best of our knowledge, Theorems 8 and 9 are the only existing method for checking whether a vector-valued stochastic processes is a weak or a very-weak martingale. The closest previous procedure derives the distributions of test statistics required for testing the null hypothesis that a given univariate stochastic process is a very-weak martingale (Park & Whang, 2005). To apply it to our case, for every $l \in \{1,\dots,L\}$, we would need to assume that the entries of $\nu^{(l-1)}(X)$ are independent; we would need to assume the same for $Y^{(l-1)}$. This is an extremely strong assumption and unlikely to hold in practice.

Remark 10. *For Theorem 5 and Corollary 6 to hold it is enough that $\boldsymbol{\nu}$ and $\mathbf{Y}$ are a very-weak martingales (Hayes, 2005, Theorem 1.8). However, we require them to be weak martingales for the following result to hold. Pick any $l \in \{1,\dots,L-1\}$. Then, we have that*

$$\begin{aligned}
&\mathbb{E}_{(X,Y)\sim\mathscr{D}}\left[\nu(X) - \mathbb{E}_{(X,Y)\sim\mathscr{D}}\left(\nu(X)\right) \mid \nu^{(l)}(X) - \mathbb{E}_{(X,Y)\sim\mathscr{D}}\left(\nu^{(l)}(X)\right)\right] \\
&= \mathbb{E}_{(X,Y)\sim\mathscr{D}}\left[\nu(X) \mid \nu^{(l)}(X) - \mathbb{E}_{(X,Y)\sim\mathscr{D}}\left(\nu^{(l)}(X)\right)\right] - \mathbb{E}_{(X,Y)\sim\mathscr{D}}\left[\nu(X)\right] \\
&= \nu^{(l)}(X) - \mathbb{E}_{(X,Y)\sim\mathscr{D}}\left[\nu^{(l)}(X)\right],
\end{aligned} \tag{6}$$

where the last equality comes from the weak martingale property of $\mathbf{Y}$. Then, the equalities in equation 6 imply that, for all $a > 0$,

$$\begin{aligned}
&P_{(X,Y)\sim\mathscr{D}}\left(\|\mathbb{E}_{(X,Y)\sim\mathscr{D}}\left[\nu(X) \mid \nu^{(l)}(X) - \mathbb{E}_{(X,Y)\sim\mathscr{D}}\left(\nu^{(l)}(X)\right)\right] - \mathbb{E}_{(X,Y)\sim\mathscr{D}}\left[\nu(X)\right]\|_2 \geq Ma\right) \\
&= P_{(X,Y)\sim\mathscr{D}}\left(\|\nu^{(l)}(X) - \mathbb{E}_{(X,Y)\sim\mathscr{D}}\left[\nu^{(l)}(X)\right]\|_2 \geq Ma\right) < 2\exp\left(\frac{1-(Ma-1)^2}{2l}\right),
\end{aligned} \tag{7}$$

where the inequality comes from Corollary 6. Equation equation 7 tells us that we can approximate arbitrarily well the value $\mathbb{E}_{(X,Y)\sim\mathscr{D}}[\nu(X)]$ that the output $\nu(X)$ of our SDNN concentrates around with the quantity $\mathbb{E}_{(X,Y)\sim\mathscr{D}}[\nu(X) \mid \nu^{(l)}(X) - \mathbb{E}_{(X,Y)\sim\mathscr{D}}(\nu^{(l)}(X))]$. This approximation improves as we move towards the final layer.

2.3 Classification accuracy

In this section, we focus on binary classification for the sake of exposition. As we pointed out earlier, a DNN $\nu : \mathbb{R}^d \to \mathbb{R}^p$, together with a choice of score function $s : \mathbb{R}^p \to \mathbb{R}$, gives us a classifier. If the score function s computed at the output value $\nu(x)$, $s(\nu(x))$, exceeds some decision threshold c, then the neural network predicts x is from a certain class $\mathscr{C}_1$, otherwise x is from the other category $\mathscr{C}_2$. The input space is then partitioned into two disjoint subsets by the decision boundary $\mathcal{B} := \{x \in \mathbb{R}^d : \nu(x) = s^{-1}(c)\}$. We call connected regions with value above the threshold *postive regions*, while those having value below the threshold *negative regions*.

For the sake of exposition suppose that $n_L = 1$, that is, $\nu : \mathbb{R}^d \to \mathbb{R}$. Then, in our stochastic setting, we have that $s(\nu(X))$ is a random variable that maps into $\mathbb{R}$; we assume it has finite first moment. We have the following important result.

Proposition 11. *If $a \leq s(\nu(X)) \leq b$ with probability 1, for some $a, b \in \mathbb{R}$, $a < b$, and $c \in [a, b]$, then for all $t > 0$,*

$$P_{(X,Y)\sim\mathscr{D}}\left(\left|s(\nu(X)) - \mathbb{E}_{(X,Y)\sim\mathscr{D}}\left[s(\nu(X))\right]\right| \geq t\right) \leq 2\exp\left(-\frac{2t^2}{(b-a)^2}\right).$$

Proposition 11 tells us that $s(\nu(X))$ concentrates around its expected value. The fact that $s(\nu(X))$ is bounded is a mild condition, as long as $c \in [a, b]$; it simply amounts to the choice of the score function.

We now consider the *expected classifier* based on the *expected decision boundary* (EDB)

$$\mathcal{B}_{exp} := \left\{x \in \mathbb{R}^d : \mathbb{E}_{(X,Y)\sim\mathscr{D}}\left[s(\nu(X))\right] = c\right\},$$

for some decision threshold c. That is,

(i) if $\mathbb{E}_{(X,Y)\sim\mathscr{D}}[s(\nu(X))] > c$, then $x \in \mathscr{C}_1$;

(ii) if $\mathbb{E}_{(X,Y)\sim\mathscr{D}}[s(\nu(X))] < c$, then $x \in \mathscr{C}_2$.

Note that being based on the value $\mathbb{E}_{(X,Y)\sim\mathscr{D}}[s(\nu(X))]$, the expected classifier will most likely not be perfect. We can provide a probabilistic bound for the classification error of the expected classifier.

Proposition 12. *If $\mathbb{E}_{(X,Y)\sim\mathscr{D}}[s(\nu(X))] > c$, then there exists $t_1 > 0$ such that*

$$P_{(X,Y)\sim\mathscr{D}}\left(s(\nu(X)) \leq c\right) \leq \exp\left(-\frac{2{t_1}^2}{(b-a)^2}\right). \tag{8}$$

If instead $\mathbb{E}_{(X,Y)\sim\mathscr{D}}[s(\nu(X))] < c$, then there exists $t_2 > 0$, possibly different than t_1, such that

$$P_{(X,Y)\sim\mathscr{D}}\left(s(\nu(X)) \geq c\right) \leq \exp\left(-\frac{2{t_2}^2}{(b-a)^2}\right). \tag{9}$$

Call p_1 the bound in equation 8 and p_2 the one in equation 9. Then, Proposition 12 tells us that the expected classifier is correct with probability $p > 1 - \max\{p_1, p_2\}$.

3 Optimal number of layers

We find the number of layers for a generic SDNN that strikes the perfect balance between accuracy of the analysis and computational cost. We do so using an optimal stopping technique. We assume that ν has constant width p for the same reasons as in section 2.2.

Let $\mathcal{B}(\mathbb{R}^p)$ be the Borel σ-algebra on $\mathbb{R}^p$, and consider a collection $(\xi^{(l)})_{l=0}^L$ of $\mathcal{B}(\mathbb{R}^p)$-measurable functions $\xi^{(l)} : \mathbb{R}^p \to \mathbb{R}$. Then, $\boldsymbol{\nu}$ induces the stochastic process $\boldsymbol{\gamma} = (\gamma^{(l)})_{l=0}^L$, where $\gamma^{(l)}$ is a random variable defined as $\gamma^{(l)} = \xi^{(l)} \circ \nu^{(l)}(X) : \Omega \to \mathbb{R}$, that is, $\gamma^{(l)} = \xi^{(l)}(\nu^{(l)}(X))$, for every l. We interpret $\gamma^{(l)}$ as the utility

of choosing l many layers for the neural network, that is, of stopping the observation of $\boldsymbol{\gamma}$ at layer l. For example, we can select

$$\gamma^{(l)} = \xi^{(l)}(\nu^{(l)}(X)) \equiv \xi_c^{(l)}(\nu^{(l)}(X)) = \begin{cases} 0 & \text{if } l = 0, \\ \frac{\left[\mathrm{LOSS}\left(\nu^{(l)}(X), Y^\star\right)\right]^{-1}}{c\sqrt{l}} & \text{if } l \in \{1, \dots, L\} \end{cases},$$

for some $c > 0$, where $\mathrm{LOSS}(\cdot, \cdot)$ is the loss function we use to train the neural network, and $Y^\star$ is the correct response to input X. This is a reasonable choice because $\gamma^{(l)}$ is higher the "closer" $\nu^{(l)}(X)$ is to $Y^\star$ (in the sense that $\mathrm{LOSS}(\nu^{(l)}(X), Y^\star)$ is small), but at the same time penalizes values of l that are too large. The optimal number of layers gives us the balance between cost and accuracy that we are looking for.

Then, we consider the natural filtration $\mathbf{F}^{\boldsymbol{\gamma}}$ of $\mathcal{F}$ with respect to $\boldsymbol{\gamma}$ given by

$$\mathcal{F}_l^{\boldsymbol{\gamma}} := \sigma\left(\left\{\left(\gamma^{(s)}\right)^{-1}(A) : s \in \mathbb{N}_0, s \leq l, A \in \mathcal{B}(\mathbb{R})\right\}\right),$$

where $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$.[1] We assume that $\boldsymbol{\gamma}$ is adapted to filtration $\mathbf{F}^{\boldsymbol{\gamma}}$, that is, $\gamma^{(l)}$ is $\mathcal{F}_l^{\boldsymbol{\gamma}}$-measurable, for all l. We interpret $\mathcal{F}_l^{\boldsymbol{\gamma}}$ as the information available up to layer l. Our decision regarding whether to choose l many layers – that is, to stop observing $\boldsymbol{\gamma}$ at layer l – must be based on this information only (no anticipation is allowed).

Definition 13. *Given a generic filtered probability space $(\Omega, \mathcal{F}, (\mathcal{F}_l)_{l\in\mathbb{N}_0}, P)$, a random variable $\tau : \Omega \to \mathbb{N}_0 \cup \{\infty\}$ is called a Markov time if $\{\tau \leq l\} \in \mathcal{F}_l$, for all $l \in \mathbb{N}_0$. A Markov time is called a stopping time if $\tau < \infty$ P-a.s.*

We denote the family of all stopping times by $\mathfrak{M}$, and the family of all Markov times by $\overline{\mathfrak{M}}$. A family that we are going to use later in this section is $\mathfrak{M}_l^L := \{\tau \in \mathfrak{M} : l \leq \tau \leq L\}$, $0 \leq l \leq L$. For notational convenience, we let $\mathfrak{M}^L \equiv \mathfrak{M}_0^L$.

The optimal stopping problem we study is the following

$$V_\star = \sup_\tau \mathbb{E}_{(X,Y)\sim\mathscr{D}}\left[\gamma^{(\tau)}\right] \tag{10}$$

where the supremum is taken over a family of stopping times. Note that equation 10 involves two tasks, that is computing the value function $V_\star$ as explicitly as possible, and finding an optimal stopping time $\tau_\star$ at which the supremum is attained.

To ensure that the expected value in equation 10 exists, we need a further assumption, that is

$$\mathbb{E}_{(X,Y)\sim\mathscr{D}}\left[\sup_{l\leq k\leq L}\left|\gamma^{(k)}\right|\right] < \infty, \tag{11}$$

If equation 11 is satisfied, then $\mathbb{E}_{(X,Y)\sim\mathscr{D}}[\gamma^{(\tau)}(x)]$ is well defined for all $\tau \in \mathfrak{M}_l^L$.

To each of the families $\mathfrak{M}_l^L$ we assign the following value function

$$V_l^L := \sup_{\tau\in\mathfrak{M}_l^L} \mathbb{E}_{(X,Y)\sim\mathscr{D}}\left[\gamma^{(\tau)}\right], \tag{12}$$

$0 \leq l \leq L$. For notational convenience, we let $V^L \equiv V_0^L$.

We solve problem equation 12 using the backward induction approach outlined in Peskir & Shiryaev (2006, Section 1.2). We assume $L < \infty$, but this is without loss of generality. Notice that equation 12 can be rewritten as

$$V_l^L = \sup_{l\leq\tau\leq L} \mathbb{E}_{(X,Y)\sim\mathscr{D}}\left[\gamma^{(\tau)}\right], \tag{13}$$

[1] For the definitions of filtration and natural filtration, see Appendix B.

where τ is a stopping time and $0 \leq l \leq L$. We solve the problem by letting time go backwards; we proceed recursively. Let L be a high number in $\mathbb{N}$, e.g. 1000. It is going to represent the maximum number of layers we deem "usable" for our neural network. We consider an ancillary sequence of random variables $(S_l^L)_{0\leq l\leq L}$ induced by $\boldsymbol{\gamma}$ that is built as follows. For $l = L$ we stop, and our utility is $S_L^L = \gamma^{(L)}$; for $l = L-1$, we can either stop or continue. If we stop, our utility is $S_{L-1}^L = \gamma^{(L-1)}$, while if we continue our utility is $S_{L-1}^L = \mathbb{E}_{(X,Y)\sim\mathscr{D}}[S_L^L \mid \mathcal{F}_{L-1}^{\boldsymbol{\gamma}}]$. As it is clear from the latter conclusion, our decision about stopping at layer $l = L-1$ or continuing with an extra layer must be based on the information contained in $\mathcal{F}_{L-1}^{\boldsymbol{\gamma}}$ only. So, if $\gamma^{(L-1)} \geq \mathbb{E}_{(X,Y)\sim\mathscr{D}}[S_L^L \mid \mathcal{F}_{L-1}^{\boldsymbol{\gamma}}]$, we stop at layer $L-1$, otherwise we add an extra layer. For $l \in \{L-2, \ldots, 0\}$ the considerations are continued analogously.

By the backward induction method we just described, we have that the elements of the sequence $(S_l^L)_{0\leq l\leq L}$ are defined recursively as

$$S_l^L = \gamma^{(L)}, \quad \text{for } l = L, \tag{14}$$

$$S_l^L = \max\left\{\gamma^{(l)}, \mathbb{E}_{(X,Y)\sim\mathscr{D}}\left[S_{l+1}^L \mid \mathcal{F}_l^{\boldsymbol{\gamma}}\right]\right\}, \quad \text{for } l \in \{L-1, \ldots, 0\}. \tag{15}$$

The method also suggests that we consider the following stopping time

$$\tau_l^L = \inf_{l\leq k\leq L}\left\{S_k^L = \gamma^{(k)}\right\}, \tag{16}$$

for all $0 \leq l \leq L$. Notice that the infimum is always attained. The following is the main result of this section; it tells us that τ_l^L is indeed the optimal stopping time for problem equation 13. It comes from Peskir & Shiryaev (2006, Theorem 1.2).

Theorem 14. *Consider the optimal stopping problem equation 13, and assume that equation 11 holds. Then, for all $l \in \{0, \ldots, L\}$, we have that*

$$S_l^L \geq \mathbb{E}_{(X,Y)\sim\mathscr{D}}\left[\gamma^{(\tau)} \mid \mathcal{F}_l^{\boldsymbol{\gamma}}\right], \quad \textit{for all } \tau \in \mathfrak{M}_l^L, \tag{17}$$

$$S_l^L = \mathbb{E}_{(X,Y)\sim\mathscr{D}}\left[\gamma^{(\tau_l^L)} \mid \mathcal{F}_l^{\boldsymbol{\gamma}}\right]. \tag{18}$$

In addition, fix any $l \in \{0, \ldots, L\}$. Then,

(i) the stopping time τ_l^L is optimal for equation 13;

(ii) if $\tau_\star$ is any optimal stopping time for equation 13, then $\tau_l^L \leq \tau_\star$ $P_{(X,Y)\sim\mathscr{D}}$-a.s.;

(iii) The stopped sequence $(S_{k\wedge\tau_l^L}^L)_{l\leq k\leq L}$ is a strong martingale.

It follows immediately from Theorem 14.(i) that the optimal number of layers for our neural network ν is given by τ_1^L.

4 Application: stochastic feedforward neural networks with ReLU activations

In this section we apply our results for a general SDNN to a stochastic version of the feedforward neural network with ReLU activation (FNNRA) introduced in Zhang et al. (2018). The key insight in Zhang et al. (2018) is that ideas from tropical algebra can be used to study feedforward neural networks, especially with ReLU activations. The intuition is that the activation function of a feedforward neural network requires computing a maximum, which turns out to correspond to tropical addition.

We first present a short summary of ideas from tropical algebra that will be used and notation for feedforward neural networks.

4.1 Tropical algebra

A detailed introduction to tropical algebra and tropical geometry is provided in Itenberg et al. (2009); Maclagan & Sturmfels (2015). The fundamental element of tropical algebra, the tropical semiring, is given by

$$\mathbb{T} = (\mathbb{R} \cup \{-\infty\}, \oplus, \odot).$$

The two operations $\oplus$ and $\odot$ are called *tropical addition* and *tropical multiplication*, respectively, and are such that $a \oplus b := \max\{a, b\}$ and $a \odot b := a + b$, for all $a, b \in \mathbb{R} \cup \{-\infty\}$. The distributive law holds for tropical addition and multiplication, the identity element of tropical addition is $-\infty$, and the identity element of tropical multiplication is 0. The tropical semiring is idempotent in the sense that $a \oplus a \oplus \cdots \oplus a = a$, for all $a \in \mathbb{R} \cup \{-\infty\}$. Because of this, there is no tropical subtraction, but tropical division is well defined

$$a \oslash b := a - b,$$

for all $a, b \in \mathbb{R} \cup \{-\infty\}$. Tropical exponentiation is well defined as well, for all $a \in \mathbb{R} \cup \{-\infty\}$, we have that

$$a^{\odot b} := \begin{cases} a \cdot b & \text{if } b \in \mathbb{Z}_+ \\ (-a) \cdot (-b) & \text{if } b \in \mathbb{Z}_- \end{cases}.$$

As we can see, tropical exponentiation is well defined only for integer exponents. We also have that

$$-\infty^{\odot a} := \begin{cases} -\infty & \text{if } a > 0 \\ 0 & \text{if } a = 0 \\ \text{undefined} & \text{if } a < 0 \end{cases}.$$

Notice that the tropical semiring is a (non-Diophantine) abstract prearithmetic, where the partial order is defined on the extended reals (Burgin & Czachor, 2020; Caprio et al., 2021). We can now define tropical polynomials and tropical rational functions.

A *tropical monomial* in d variables $x_1, \ldots, x_d$ is an expression of the form

$$c \odot x_1^{\odot a_1} \odot x_2^{\odot a_2} \odot \cdots \odot x_d^{\odot a_d},$$

where $c \in \mathbb{R} \cup \{-\infty\}$ and $a_1, \ldots, a_d \in \mathbb{N}$. As a notational shorthand, we can write it in multi-index notation as cx^α, where $\alpha = (a_1, \ldots, a_d)^\top \in \mathbb{N}^d$ and $x = (x_1, \ldots, x_d)^\top$.

A *tropical polynomial* $f(x) = f(x_1, \ldots, x_d)$ is a finite tropical sum of tropical monomials,

$$f(x) = c_1 x^{\alpha_1} \oplus \cdots \oplus c_r x^{\alpha_r},$$

where $\alpha_i = (a_{i1}, \ldots, a_{id})^\top \in \mathbb{N}^d$ and $c_i \in \mathbb{R} \cup \{-\infty\}$, $i \in \{1, \ldots, r\}$. We assume that $\alpha_i \neq \alpha_j$, for all $i \neq j$.

A *tropical rational function* is a standard difference, that is, a tropical quotient of two tropical polynomials $f(x)$ and $g(x)$,

$$f(x) - g(x) = f(x) \oslash g(x).$$

We denote a tropical rational function by $f \oslash g$, where f and g are tropical polynomial functions. A tropical polynomial f can be seen as a tropical rational function, indeed $f = f \oslash 0$. Hence, any result holding for tropical rational functions hold also for tropical polynomials.

A d-variate tropical polynomial $f(x)$ defines a function $f : \mathbb{R}^d \to \mathbb{R}$ that is a convex function, in that taking max and sum of convex functions preserves convexity (Boyd & Vandenberghe, 2004). So, a tropical rational function $f \oslash g : \mathbb{R}^d \to \mathbb{R}$ is a difference of convex function (An & Tao, 2005; Hartman, 1959).

A function $F : \mathbb{R}^d \to \mathbb{R}^p$, $x = (x_1, \ldots, x_d)^\top \mapsto (f_1(x), \ldots, f_p(x))^\top$, is called a *tropical polynomial map* if each $f_i : \mathbb{R}^d \to \mathbb{R}$ is a tropical polynomial, for all $i \in \{1, \ldots, p\}$, and a *tropical rational map* if $f_1(x), \ldots, f_p(x)$ are tropical rational functions.

Tropical polynomials and tropical rational functions are piecewise linear functions. Hence, a tropical rational map is a piecewise linear map and the notion of a linear region applies. A *linear region* of a tropical rational map F is a maximal connected subset of the domain on which F is linear. The number of linear regions of F is denoted by $\mathcal{N}(F)$.

4.2 Deterministic feedforward neural networks

We use this section to fix the notation for feedforward neural networks. We then introduce the the FNNRA proposed in Zhang et al. (2018). We restrict our attention to fully connected feedforward neural networks.

An L-layered feedforward neural network is a map $\nu : \mathbb{R}^d \to \mathbb{R}^p$ given by a composition of functions

$$\nu \equiv \nu^{(L)} = \sigma^{(L)} \circ \rho^{(L)} \circ \sigma^{(L-1)} \circ \rho^{(L-1)} \circ \cdots \circ \sigma^{(1)} \circ \rho^{(1)}.$$

The *preactivation functions* $\rho^{(1)}, \ldots, \rho^{(L)}$ are affine transformations to be determined, and the *activation functions* $\sigma^{(1)}, \ldots, \sigma^{(L)}$ are chosen and fixed in advance. Affine function $\rho^{(l)} : \mathbb{R}^{n_{l-1}} \to \mathbb{R}^{n_l}$ is given by a weight matrix $A^{(l)} \in \mathbb{Z}^{n_l \times n_{l-1}}$ and a bias vector $b^{(l)} \in \mathbb{R}^{n_l}$,

$$\rho^{(l)}(\nu^{(l-1)}) := A^{(l)}\nu^{(l-1)} + b^{(l)}.$$

The (i,j)-th coordinate of $A^{(l)}$ is denoted by $a_{ij}^{(l)}$, and the i-th coordinate of $b^{(l)}$ by $b_i^{(l)}$. Collectively they form the *parameters* of the l-th layer. Notice that, for all l, $A^{(l)}$ can be decomposed as a difference of two nonnegative integer valued matrices, $A^{(l)} = A_+^{(l)} - A_-^{(l)}$, with $A_+^{(l)}, A_-^{(l)} \in \mathbb{Z}^{n_l \times n_{l-1}}$, so that their entries are

$$a_{+ij}^{(l)} = \max\{a_{ij}, 0\}, \quad a_{-ij}^{(l)} = \max\{-a_{ij}, 0\}.$$

For a vector input $x \in \mathbb{R}^{n_l}$, $\sigma^{(l)}(x)$ is understood to be in coordinatewise sense. We make the following assumptions on the architecture of our feedforward neural network:

(a) the weight matrices $A^{(1)}, \ldots, A^{(L)}$ are integer-valued;

(b) the bias vectors $b^{(1)}, \ldots, b^{(L)}$ are real-valued;

(c) the activation functions $\sigma^{(1)}, \ldots, \sigma^{(L)}$ take the form

$$\sigma^{(l)}(x) := \max\{x, t^{(l)}\} = \left(\max\{x_1, t_1^{(l)}\}, \ldots, \max\{x_{n_l}, t_{n_l}^{(l)}\}\right)^\top,$$

where $t^{(l)} \in (\mathbb{R} \cup \{-\infty\})^{n_l}$ is the *threshold vector*, and x_j and $t_j^{(l)}$ denote the j-th element of x and $t^{(l)}$, respectively.

We assume all the neural networks in this section to satisfy (a)-(c).

Assumption (b) is general, and yields no loss of generality. The same goes for (a), since:

- real weights can be approximated arbitrarily closely by rational weights;
- one may "clear denominators" in these rational weights by multiplying them by the least common multiple of their denominators to obtain integer weights;
- scaling all weights and biases by the same positive constant does not influence the workings of a neural network.

The form of the activation function in (c) includes both ReLU activation ($t^{(l)} = 0\mathbf{1}_{n_l}$) and identity map ($t^{(l)} = -\infty\mathbf{1}_{n_l}$, so that $\sigma^{(l)}(x) = x$) as special cases, where $\mathbf{1}_{n_l}$ is vector $(1, \ldots, 1)^\top$ having n_l entries. We only consider the ReLU activation function in this section as we are generalizing the model in Zhang et al. (2018) where the tropical algebra makes the most sense for ReLU activations. In addition, much of the theory literature on neural networks has focused on ReLU networks (Arora et al., 2018; Montúfar et al., 2014; Zhang et al., 2018).

Let us now describe the the deterministic ReLU neural network proposed in Zhang et al. (2018); its architecture is depicted in Figure 1.[2] Zhang et al. (2018, Section 5) state that the $(l+1)$-th layer $\nu^{(l+1)}(x)$ of an L-layered FNNRA $\nu(x)$ can be written as $\nu^{(l+1)}(x) = F^{(l+1)}(x) - G^{(l+1)}(x)$, for all $x \in \mathbb{R}^d$, where

$$\begin{aligned} F^{(l+1)}(x) &= H^{(l+1)}(x) \oplus G^{(l+1)}(x) \odot t^{(l+1)} = \max\{H^{(l+1)}(x), G^{(l+1)}(x) + t^{(l+1)}\}, \\ G^{(l+1)}(x) &= A_+^{(l+1)} G^{(l)}(x) \odot A_-^{(l+1)} F^{(l)}(x) = A_+^{(l+1)} G^{(l)}(x) + A_-^{(l+1)} F^{(l)}(x), \\ H^{(l+1)}(x) &= A_+^{(l+1)} F^{(l)}(x) \odot A_-^{(l+1)} G^{(l)}(x) \odot b^{(l+1)} = A_+^{(l+1)} F^{(l)}(x) + A_-^{(l+1)} G^{(l)}(x) + b^{(l+1)}, \end{aligned} \tag{19}$$

where $F^{(l)}(x)$ and $G^{(l)}(x)$ are vectors in $\mathbb{R}^{n_l}$ whose coordinates are tropical polynomials in x.[3] That is,

$$\begin{aligned} F^{(l)}(x) &= \left(f_1^{(l)}(x), \ldots, f_{n_l}^{(l)}(x)\right)^\top, \\ G^{(l)}(x) &= \left(g_1^{(l)}(x), \ldots, g_{n_l}^{(l)}(x)\right)^\top, \end{aligned} \tag{20}$$

where

$$\begin{aligned} f_j^{(l)}(x) &= \max\{c_{j1} x^{\odot \alpha_{j1}}, \ldots, c_{jr} x^{\odot \alpha_{jr}}\}, \\ g_j^{(l)}(x) &= \max\{c'_{j1} x^{\odot \alpha'_{j1}}, \ldots, c'_{jr} x^{\odot \alpha'_{jr}}\}, \end{aligned} \tag{21}$$

for all $j \in \{1, \ldots, n_l\}$ and some $r \in \mathbb{N}$, and

$$c_{js} x^{\odot \alpha_{js}} := c_{js} + \alpha_{js,1} x_1 + \alpha_{js,2} x_2 + \ldots + \alpha_{js,d} x_d,$$

for all $j \in \{1, \ldots, n_l\}$ and all $s \in \{1, \ldots, r\}$. Similarly,

$$c'_{js} x^{\odot \alpha'_{js}} := c'_{js} + \alpha'_{js,1} x_1 + \alpha'_{js,2} x_2 + \ldots + \alpha'_{js,d} x_d,$$

for all $j \in \{1, \ldots, n_l\}$ and all $s \in \{1, \ldots, r\}$. Notice that $c_{js}, c'_{js} \in \mathbb{R}$ for all j and all s, and that $\alpha_{js} = (\alpha_{js,1}, \ldots, \alpha_{js,d})^\top, \alpha'_{js} = (\alpha'_{js,1}, \ldots, \alpha'_{js,d})^\top \in \mathbb{N}^d$, for all j and all s.

4.3 Stochastic feedforward ReLU networks

In this section we derive a stochastic version of the FNNRA described in section 4.2. We introduce two sources of of stochasticity: (a) the initialization or the starting values of the neural network are random (that is, $F^{(0)}(X)$ and $G^{(0)}(X)$ are random vectors), and (b) the parameters of all the layers are aleatory (that is, $A^{(l)}$ is a random matrix with integer entries, and $b^{(l)}$ is a random vector with real entries, for all $l \in \{1, \ldots, L\}$). We now state the update rules for the first layer, and the second layer given the first. By induction, this is enough to specify the stochastic feedforward network.

For the first layer we sample random vectors $F^{(0)}(X)$ and $G^{(0)}(X)$ from a distribution $\mathscr{R}$ on $\mathbb{R}^{n_0}$ that is built as follows. Sample

$$(c_{j1}, \ldots, c_{jr})^\top, (c'_{j1}, \ldots, c'_{jr})^\top \sim \mathscr{S}_j \text{ on } \mathbb{R}^r, \quad \alpha_{j1}, \ldots, \alpha_{jr}, \alpha'_{j1}, \ldots, \alpha'_{jr} \sim \mathscr{T}_j \text{ on } \mathbb{N}^d,$$

for $j \in \{1, \ldots, n_0\}$. Note that we do not require the α_{js}'s and the α'_{js}'s to be iid and we do not require the samples to be independent across the index j. Then, $F^{(0)}(x)$ and $G^{(0)}(x)$ – the realizations of $F^{(0)}(X)$ and $G^{(0)}(X)$, respectively – are computed according to equation 20 and equation 21. Notice that $F^{(0)}(X)$ and $G^{(0)}(X)$ need not be independent.

[2] Figure 1 is a replica of Zhang et al. (2018, Figure A.1).

[3] Notice that we can write the i-th entry of vector $A_+^{(l+1)} G^{(l)}(x)$ in tropical notation as

$$\bigodot_{j \in \{1, \ldots, n_l\}} \left[a_{+ij}^{(l+1)}\right]^{\odot g_j^{(l)}(x)}.$$

We can write similarly the i-th entries of $A_-^{(l+1)} G^{(l)}(x)$, $A_+^{(l+1)} F^{(l)}(x)$, and $A_-^{(l+1)} F^{(l)}(x)$.

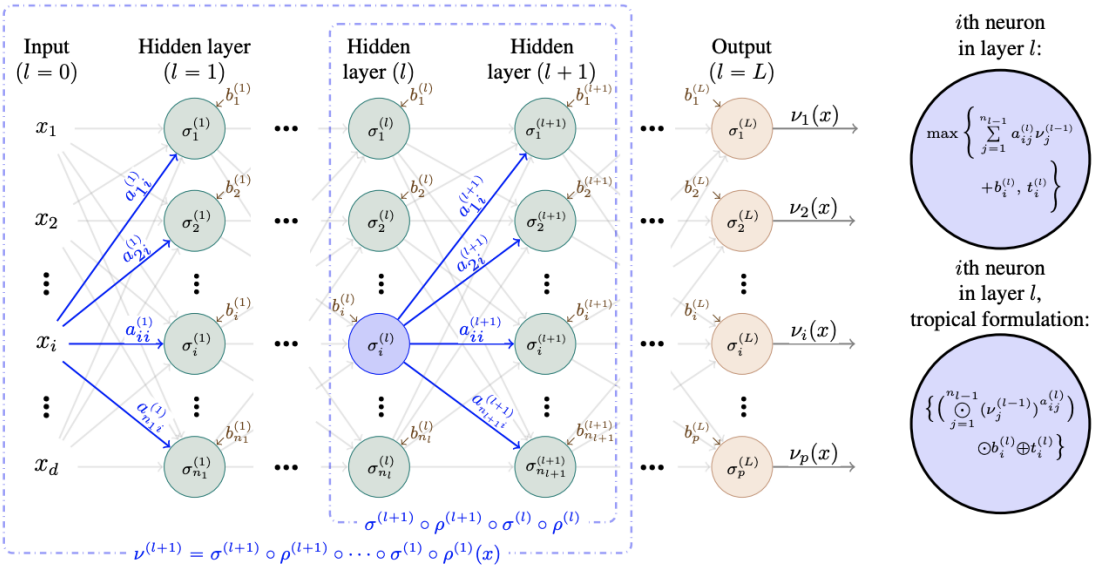

Figure 1: Architecture of the FNNRA $\nu : \mathbb{R}^d \to \mathbb{R}^p$ with L layers proposed in Zhang et al. (2018).

To compute the subsequent layer we first sample $A^{(l+1)} \sim \mathscr{P}$ on $\mathbb{Z}^{n_{l+1} \times n_l}$ and $b^{(l+1)} \sim \mathscr{Q}$ on $\mathbb{R}^{n_{l+1}}$, for $l \in \{0, \dots, L-1\}$. After computing $F^{(l+1)}(x)$ and $G^{(l+1)}(x)$ from $F^{(l)}(x)$ and $G^{(l)}(x)$ by equation equation 19, we find $\nu^{(l+1)}(x) = F^{(l)}(x) - G^{(l)}(x)$. Note that $\nu^{(l+1)}(x)$ is the realization of random vector $\nu^{(l+1)}(X) : \Omega \to \mathbb{R}^{n_{l+1}}$ whose elements may be correlated and that we assume to have finite fist moment. We now apply Corollary 4 to our stochastic FNNRA.

Proposition 15. *If $\mathscr{P}$, $\mathscr{Q}$, $\mathscr{S}_j$, and $\mathscr{T}_j$, $j \in \{1, \dots, n_0\}$ are bounded, then for all $l \in \{1, \dots, L\}$, there exists $\sigma_{(l)} \in \mathbb{R}$ such that $\|\nu^{(l)}(X)\|_2 \leq \sigma_{(l)}$ and equation 3 holds.*

The assumption that $\mathscr{P}$, $\mathscr{Q}$, $\mathscr{S}_j$, and $\mathscr{T}_j$, $j \in \{1, \dots, n_0\}$, are bounded should always be verified: even if the distributions we want to use are unbounded, we can always truncate them and use the truncated versions to obtain the concentration result in Proposition 15.

Remark 16. *Notice that if $t^{(l)}$ were a random quantity distributed according to $\mathscr{V}$ on $\mathbb{R}^{n_l}$, and possibly correlated with other $t^{(k)}$'s, Proposition 15 would still hold, provided that $\mathscr{V}$ is bounded.*[4]

4.4 Concentration inequality for positive and negative regions

Recall that tropical polynomials are piecewise linear, so the notion of linear regions applies. A linear region of a tropical polynomial f is a maximal connected subset of the domain on which f is linear. The number of linear regions of f is denoted by $\mathcal{N}(f)$. Zhang et al. (2018) use this notion to provide bounds on the number of positive and negative regions the neural network divides the input space.

Proposition 17. *(Zhang et al., 2018, Proposition 6.1) Let ν be an L-layer stochastic neural network satisfying (a)-(c) in section 4.2 such that $t^{(L)} = -\infty$ and $p = n_L = 1$, that is, $\nu : \mathbb{R}^d \to \mathbb{R}$. Let the score function $s : \mathbb{R} \to \mathbb{R}$ be injective with decision threshold c in its range. If $\nu = f \oslash g$, where f and g are tropical polynomials, then the number of connected positive regions is at most $\mathcal{N}(f)$, while the number of connected negative regions is at most $\mathcal{N}(g)$.*

Suppose now all the assumptions of Proposition 17 hold. In our stochastic setting, we have that f and g are both random variables, so $\mathcal{N}(f)$ and $\mathcal{N}(g)$ are two stochastic quantities in $\mathbb{N}$. Assume their first moment is finite and notice that they are bounded below by 0. Then, the following proposition holds.

[4] If $t^{(l)} \sim \mathscr{V}$ on $\mathbb{R}^{n_l}$, this means that we rule out the identity activation map. To include it, $\mathscr{V}$ should be a distribution on $(\mathbb{R} \cup \{-\infty\})^{n_l}$.

Proposition 18. *If $\mathcal{N}(f) \leq b_1$ and $\mathcal{N}(g) \leq b_2$ with probability 1, for some possibly different $b_1, b_2 \in \mathbb{N}$, for all $t > 0$,*

$$P_{(X,Y)\sim\mathscr{D}}\left(\left|\mathcal{N}(f) - \mathbb{E}_{(X,Y)\sim\mathscr{D}}[\mathcal{N}(f)]\right| \geq t\right) \leq 2\exp\left(-\frac{2t^2}{{b_1}^2}\right)$$

and

$$P_{(X,Y)\sim\mathscr{D}}\left(\left|\mathcal{N}(g) - \mathbb{E}_{(X,Y)\sim\mathscr{D}}[\mathcal{N}(g)]\right| \geq t\right) \leq 2\exp\left(-\frac{2t^2}{{b_2}^2}\right).$$

The assumption that $\mathcal{N}(f)$ and $\mathcal{N}(g)$ have an upper bound is always verified: all the possible realizations of f and g have a finite number of linear regions. Proposition 18 tells us that the upper bounds for the number of positive and negative regions concentrate around their expected value.

5 Conclusion

In this paper we present concentration inequalities for the hidden layers and the output of a generic SDNN based on the ideas of norm-subGaussian distribution and of weak martingale. We introduce the notion of expected classifier and give a probabilistic bound to its classification error. We also find – via an optimal stopping procedure – the number of layers for the SDNN that strikes the perfect balance between computational cost and accuracy of the analysis. Finally, we apply our findings to a stochastic version of the FNNRA of Zhang et al. (2018). In future work, we plan to explore the geometric properties of SDNNs, in the spirit of Alfarra et al. (2021); Hauser & Ray (2017); Maragos et al. (2021).

References

Motasem Alfarra, Adel Bibi, Hasan Hammoud, Mohamed Gaafar, and Bernard Ghanem. On the decision boundaries of neural networks: A tropical geometry perspective. *Available at arxiv:2002.08838*, 2021.

Le Thi Hoai An and Pham Dinh Tao. The DC (difference of convex functions) programming and DCA revisited with DC models of real world nonconvex optimization problems. *Annals of Operations Research*, 133:23–46, 2005.

Raman Arora, Amitabh Basu, Poorya Mianjy, and Anirbit Mukherjee. Understanding deep neural networks with rectified linear units. In *International Conference on Learning Representations*, 2018.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *Available at arxiv:1409.0473*, 2016.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, Cambridge, 2004.

Mark Burgin and Marek Czachor. *Non-Diophantine Arithmetics in Mathematics, Physics and Psychology*. World Scientific, Singapore, 2020.

Michele Caprio, Andrea Aveni, and Sayan Mukherjee. Concerning three classes of non-Diophantine arithmetics. *Available at arxiv:2102.04197*, 2021.

Gwendoline De Bie, Gabriel Peyré, and Marco Cuturi. Stochastic deep networks. *Proceedings of Machine Learning Research*, 97:1556–1565, 2019.

Rémy Garnier and Raphaël Langhendries. Concentration inequalities for non-causal random fields. *Available at arxiv:2009.08905*, 2021.

Philip Hartman. On functions representable as a difference of convex function. *Pacific Journal of Mathematics*, 9(3):707–713, 1959.

Michael Hauser and Asok Ray. Principles of Riemannian geometry in neural networks. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Thomas P. Hayes. A large-deviation inequality for vector-valued martingales. *Submitted to Combinatorics, Probability and Computing. Available here*, 2005.

G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.

J.J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, 1982.

Chin-Wei Huang, Ahmed Touati, Pascal Vincent, Gintare Karolina Dziugaite, Alexandre Lacoste, and Aaron Courville. Stochastic neural network with Kronecker flow. *Available at arxiv:1906.04282*, 2020.

Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q. Weinberger. Deep networks with stochastic depth. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling (eds.), *Computer Vision – ECCV 2016*, pp. 646–661. Springer International Publishing, 2016.

Ilia Itenberg, Grigory Mikhalkin, and Eugenii I. Shustin. *Tropical Algebraic Geometry*, volume 35 of *Oberwolfach Seminars Book*. Birkhäuser, Basel, 2009.

Chi Jin, Praneeth Netrapalli, Rong Ge, Sham M. Kakade, and Michael I. Jordan. A short note on concentration inequalities for random vectors with subGaussian norm. *Available at arxiv:1902.03736*, 2019.

Laurent Valentin Jospin, Wray Buntine, Farid Boussaid, Hamid Laga, and Mohammed Bennamoun. Hands-on Bayesian neural networks – a tutorial for deep learning users. *Available at arXiv:2007.06823*, 2022.

Nal Kalchbrenner and Phil Blunsom. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1700–1709. Association for Computational Linguistics, 2013.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the Association for Computing Machinery*, 60(6):84–90, 2017.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521:436–444, 2015.

Kimin Lee, Jaehyung Kim, Song Chong, and Jinwoo Shin. Simplified stochastic feedforward neural networks. *Available at arxiv:1704.03188*, 2017.

Liam Li and Ameet Talwalkar. Random search and reproducibility for neural architecture search. In *Uncertainty in artificial intelligence*, pp. 367–377. PMLR, 2020.

Lanlan Liu and Jia Deng. Dynamic deep neural networks: Optimizing accuracy-efficiency trade-offs by selective execution. In Sheila A. McIlraith and Kilian Q. Weinberger (eds.), *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pp. 3675–3682. AAAI Press, 2018.

Diane Maclagan and Bernd Sturmfels. *Introduction to Tropical Geometry*, volume 161 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, 2015.

Petros Maragos, Vasileios Charisopoulos, and Emmanouil Theodosis. Tropical geometry and machine learning. *Proceedings of the IEEE*, 109(5):728–755, 2021.

Thomas Merkh and Guido Montúfar. Stochastic feedforward neural networks: Universal approximation. *Available at arxiv:1910.09763*, 2019.

Guido Montúfar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio. On the number of linear regions of deep neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems*, volume 2, pp. 2924–2932. NeurIPS, 2014.

Guilherme Ost and Patricia Reynaud-Bouret. Sparse space–time models: Concentration inequalities and Lasso. *Annales de l'Institut Henri Poincaré, Probabilités et Statistiques*, 56(4):2377–2405, 2020.

Joon Y. Park and Yoon-Jae Whang. A test of the martingale hypothesis. *Studies in Nonlinear Dynamics and Econometrics*, 9(2):1–29, 2005.

Goran Peskir and Albert N. Shiryaev. *Optimal Stopping and Free-Boundary Problems*. Lectures in Mathematics. ETH Zürich. Birkhäuser, Basel, 2006.

Mert R. Sabuncu. Intelligence plays dice: Stochasticity is essential for machine learning. *Available at arxiv:2008.07496*, 2020.

Moshe Shaked and J. George Shanthikumar. *Stochastic Orders*. Springer Series in Statistics. Springer, New York, 2007.

Charlie Tang and Russ R. Salakhutdinov. Learning stochastic feedforward neural networks. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

Matus Telgarsky. Benefits of depth in neural networks. In Vitaly Feldman, Alexander Rakhlin, and Ohad Shamir (eds.), *29th Annual Conference on Learning Theory*, volume 49 of *Proceedings of Machine Learning Research*, pp. 1517–1539, Columbia University, New York, New York, USA, 23–26 June 2016.

Andrii Trelin and Ales Prochazka. Binary stochastic filtering: a method for neural network size minimization and supervised feature selection. *Available at arxiv:1902.04510*, 2019.

Eugene Wong. Stochastic neural networks. *Algorithmica*, 6:466–478, 1991.

Sirui Xie, Hehui Zheng, Chunxiao Liu, and Liang Lin. Snas: Stochastic neural architecture search. *Available at arxiv:1812.09926*, 2020.

Liwen Zhang, Gregory Naitzat, and Lek-Heng Lim. Tropical geometry of deep neural networks. *Proceedings of Machine Learning Research*, 80:5824–5832, 2018.

Wanrong Zhu, Zhipeng Lou, and Wei Biao Wu. Beyond sub-Gaussian noises: Sharp concentration analysis for stochastic gradient descent. *Journal of Machine Learning Research*, 23(46):1–22, 2022.

A SubGaussian random variables and norm-subGaussian random vectors

SubGaussian random variables and norm-subGaussian random vectors are crucial concepts for the results in section 2.1. We first introduce the former (Jin et al., 2019).

Definition 19. *A generic random variable $X \in \mathbb{R}$ is subGaussian, written $SG(\sigma)$, if there exists $\sigma \in \mathbb{R}$ such that, for all $\theta \in \mathbb{R}$,*

$$\mathbb{E}\left[e^{\theta(X-\mathbb{E}(X))}\right] \leq \exp\left(\frac{\theta^2\sigma^2}{2}\right).$$

The following is an important characterization of subGaussian random variables.

Proposition 20. *If X is $SG(\sigma)$, then for all $t > 0$*

$$P\left(|X - \mathbb{E}(X)| \geq t\right) \leq 2\exp\left(-\frac{t^2}{2\sigma^2}\right).$$

If a random variable is bounded, then it is subGaussian.

Proposition 21. *Let X be a generic random variable in $\mathbb{R}$ such that $a \leq X \leq b$ with probability 1, for some $a, b \in \mathbb{R}$, $a < b$. Then, for all $\theta \in \mathbb{R}$,*

$$\mathbb{E}\left[e^{\theta(X-\mathbb{E}(X))}\right] \leq \exp\left(\frac{\theta^2}{2}\frac{(b-a)^2}{4}\right),$$

that is, X is $SG\left(\frac{b-a}{2}\right)$.

We now give the random vector counterpart of a subGaussian random variable.

Definition 22. *A generic random vector $X \in \mathbb{R}^d$ is norm-subGaussian, written $nSG(\sigma)$, if there exists $\sigma \in \mathbb{R}$ such that, for all $t \in \mathbb{R}$,*

$$P\left(\|X - \mathbb{E}(X)\|_2 \geq t\right) \leq 2\exp\left(-\frac{t^2}{2\sigma^2}\right).$$

Definition 22 is closely related to the characterization in Proposition 20. The following comes from Jin et al. (2019, Lemma 1), and it is the random vector counterpart of Proposition 21.

Proposition 23. *Let X be a generic bounded random vector in $\mathbb{R}^d$ such that $\|X\|_2 \leq \sigma$. Then, X is $nSG(\sigma)$.*

B Martingales and filtrations

We first define very-weak, weak, and strong martingales. They were introduced in Hayes (2005), and are extremely important for section 2.2. We denote by $\mathbf{E}$ any real Euclidean space (of finite or infinite dimension), and by $\mathbf{1_E}$ the vector $(1, 1, \ldots, 1)^\top$ having $\#\mathbf{E}$ entries, where $\#$ denotes the cardinality operator.

Definition 24. *Let $\mathbf{X} = (X_i : \Omega \to \mathbf{E})$ be a sequence of random vectors taking values in $\mathbf{E}$, such that $X_0 = 0\mathbf{1_E}$, and for every $i \geq 1$, $\mathbb{E}(\|X_i\|_2) < \infty$ and $\mathbb{E}(X_i \mid X_0, \ldots, X_{i-1}) = X_{i-1}$. Then we call $\mathbf{X}$ a strong martingale in $\mathbf{E}$.*

Definition 25. *Let $\mathbf{X} = (X_i : \Omega \to \mathbf{E})$ be a sequence of random vectors taking values in $\mathbf{E}$, such that $X_0 = 0\mathbf{1_E}$, and for every $j < i$, $\mathbb{E}(\|X_i\|_2) < \infty$ and $\mathbb{E}(X_i \mid X_j) = X_j$. Then we call $\mathbf{X}$ a weak martingale in $\mathbf{E}$.*

If $\mathbf{X}$ is a strong martingale, then it is also a weak martingale. The converse need not hold.

Definition 26. *Let $\mathbf{X} = (X_i : \Omega \to \mathbf{E})$ be a sequence of random vectors taking values in $\mathbf{E}$, such that $X_0 = 0\mathbf{1_E}$, and for every $i \geq 1$, $\mathbb{E}(\|X_i\|_2) < \infty$ and $\mathbb{E}(X_i \mid X_{i-1}) = X_{i-1}$. Then we call $\mathbf{X}$ a very-weak martingale in $\mathbf{E}$.*

If $\mathbf{X}$ is a weak martingale, then it is also a very-weak martingale. The converse need not hold. Notice that in section 2.2 we implicitly assume $\nu^{(0)}(X) = 0\mathbf{1}_p$. This is just a convention, even if it differs from the one in section 2.1 where we require $\nu^{(0)}(X)$ to be equal to X.

We then introduce the concepts of filtration and natural filtration. They are crucial for section 3.

Definition 27. *Let $(\Omega, \mathcal{F}, P)$ be a probability space and let I be an index set with a total order $\leq$. For every $i \in I$, let $\mathcal{F}_i$ be a sub-σ-algebra of $\mathcal{F}$. Then, $\mathbf{F} = (\mathcal{F}_i)_{i \in I}$ is a filtration if, for all $k \leq \ell$, $\mathcal{F}_k \subset \mathcal{F}_\ell$.*

Definition 28. *Let $(\Omega, \mathcal{F}, P)$ be a probability space and let I be an index set with a total order $\leq$. Let (S, Σ) be a measurable space and let $\mathbf{X} : I \times \Omega \to S$ be a stochastic process. Then the natural filtration of $\mathcal{F}$ with respect to $\mathbf{X}$ is defined to be the filtration $\mathbf{F}^{\mathbf{X}} = (\mathcal{F}_i^{\mathbf{X}})_{i \in I}$ given by*

$$\mathcal{F}_i^{\mathbf{X}} := \sigma\left(\left\{X_j^{-1}(A) : j \in I, j \leq i, A \in \Sigma\right\}\right).$$

That is, $\mathcal{F}_i^{\mathbf{X}}$ is the smallest σ-algebra on Ω that contains all pre-images of Σ-measurable subsets of S for "times" j up to i.

C Proofs

Proof of Theorem 3. Immediate from Definition 22 and Proposition 23. □

Proof of Corollary 4. Immediate from Theorem 3 and equation equation 2. □

Proof of Corollary 6. Immediate from Theorem 5. □

Proof of Proposition 11. Immediate from Propositions 20 and 21. □

Proof of Proposition 12. Let us first assume $\mathbb{E}_{(X,Y)\sim\mathscr{D}}[s(\nu(X))] < c$. By Proposition 11, we have that, for all $t > 0$,

$$P_{(X,Y)\sim\mathscr{D}}\left(s(\nu(X)) \geq \mathbb{E}_{(X,Y)\sim\mathscr{D}}[s(\nu(X))] + t\right) \leq \exp\left(-\frac{2t^2}{(b-a)^2}\right). \tag{22}$$

Then, since $\mathbb{E}_{(X,Y)\sim\mathscr{D}}[s(\nu(X))] < c$, there exists $t_2 > 0$ such that $\mathbb{E}_{(X,Y)\sim\mathscr{D}}[s(\nu(X))] + t_2 = c$. The bound in equation 22 implies that

$$P_{(X,Y)\sim\mathscr{D}}\left(s(\nu(X)) \geq \mathbb{E}_{(X,Y)\sim\mathscr{D}}[s(\nu(X))] + t_2\right) = P_{(X,Y)\sim\mathscr{D}}\left(s(\nu(X)) \geq c\right) \leq \exp\left(-\frac{2{t_2}^2}{(b-a)^2}\right).$$

Assume now that $\mathbb{E}_{(X,Y)\sim\mathscr{D}}[s(\nu(X))] > c$. By Proposition 11, we have that, for all $t > 0$,

$$P_{(X,Y)\sim\mathscr{D}}\left(s(\nu(X)) \leq \mathbb{E}_{(X,Y)\sim\mathscr{D}}[s(\nu(X))] - t\right) \leq \exp\left(-\frac{2t^2}{(b-a)^2}\right). \tag{23}$$

Then, since $\mathbb{E}_{(X,Y)\sim\mathscr{D}}[s(\nu(X))] > c$, there exists $t_1 > 0$ such that $\mathbb{E}_{(X,Y)\sim\mathscr{D}}[s(\nu(X))] - t_1 = c$. The bound in equation 23 implies that

$$P_{(X,Y)\sim\mathscr{D}}\left(s(\nu(X)) \leq \mathbb{E}_{(X,Y)\sim\mathscr{D}}[s(\nu(X))] - t_1\right) = P_{(X,Y)\sim\mathscr{D}}\left(s(\nu(X)) \leq c\right) \leq \exp\left(-\frac{2{t_1}^2}{(b-a)^2}\right).$$

□

Proof of Proposition 15. If $\mathscr{S}_j$ and $\mathscr{T}_j$ are bounded for all $j \in \{1, \ldots, n_0\}$, then $F^{(0)}(X)$ and $G^{(0)}(X)$ are bounded. In addition, if $\mathscr{P}$ and $\mathscr{Q}$ are bounded, then $A^{(l)}$ and $b^{(l)}$ are bounded, for all $l \in \{1, \ldots, L\}$. In turn, this entails that $\nu^{(l)}(X)$ is bounded, that is, there exists $\sigma_{(l)} \in \mathbb{R}$ such that $\|\nu^{(l)}(X)\|_2 \leq \sigma_{(l)}$, for all $l \in \{1, \ldots, L\}$. Then, the fact that equation 3 holds comes immediately from Corollary 4. □

Proof of Proposition 18. Immediate from Propositions 20 and 21. □

Proof of Proposition 20. Let X be $SG(\sigma)$. We have that, for all $s > 0$,

$$P(X - \mathbb{E}(X) \geq t) \leq P\left(e^{s(X-\mathbb{E}(X)} > e^{st}\right) \leq \frac{\mathbb{E}\left(e^{s(X-\mathbb{E}(X)}\right)}{e^{st}},$$

where the first inequality comes from using Markov's inequality, and the second one from using Chernoff's bound. Then, since X is $SG(\sigma)$, we have that

$$P(X - \mathbb{E}(X) \geq t) \leq \exp\left(\frac{s^2\sigma^2}{2} - st\right).$$

The above inequality holds for any $s > 0$ so to make it the tightest possible, we minimize with respect to $s > 0$. In particular, we solve $\phi'(s) = 0$, where $\phi(s) = \frac{s^2\sigma^2}{2} - st$, and we find $\inf_{s>0} \phi(s) = -\frac{t^2}{2\sigma^2}$. We complete the proof by repeating this process for $P(X - \mathbb{E}(X) \leq -t)$. □

Proof of Proposition 21. Without loss of generality, let $\mathbb{E}(X) = 0$. Then, let P denote the probability distribution of X. Pick any $\theta \in \mathbb{R}$ and define $\varphi(\theta) := \log \mathbb{E}_P(e^{\theta X})$. Let then Q_θ be the distribution of X defined by

$$\mathrm{d}Q_\theta(x) := \frac{e^{\theta x}}{\mathbb{E}_P(e^{\theta X})}\mathrm{d}P(x).$$

We have that

$$\varphi'(\theta) = \frac{\mathbb{E}_P(Xe^{\theta X})}{\mathbb{E}_P(e^{\theta X})} = \int x \frac{e^{\theta x}}{\mathbb{E}_P(e^{\theta X})}\mathrm{d}P(x) = \mathbb{E}_{Q_\theta}(X)$$

and that

$$\begin{aligned}\varphi''(\theta) &= \frac{\mathbb{E}_P(X^2e^{\theta X})}{\mathbb{E}_P(e^{\theta X})} - \frac{\mathbb{E}_P(Xe^{\theta X})^2}{\mathbb{E}_P(e^{\theta X})^2} \\ &= E_{Q_\theta}(X^2) - E_{Q_\theta}(X)^2 = \mathbb{V}_{Q_\theta}(X),\end{aligned}$$

where $\mathbb{V}$ denotes the variance operator. Now, by Popoviciu's inequality we have that $\mathbb{V}_{Q_\theta}(X) \leq \frac{(b-a)^2}{4}$. Then, by the fundamental theorem of calculus, we have that

$$\varphi(\theta) = \int_0^\theta \int_0^\mu \varphi''(\rho)\ \mathrm{d}\rho\ \mathrm{d}\mu \leq \frac{(b-a)^2}{4}\frac{\theta^2}{2}, \tag{24}$$

using $\varphi(0) = \log 1 = 0$ and $\varphi'(0) = \mathbb{E}_{Q_0}(X) = E_P(X) = 0$. The proof is concluded by exponentiating both sides of the inequality in equation 24. □