A Spectral Perspective of DNN Robustness to Label Noise

Oshrat Bar **Amnon Drory** **Raja Giryes**
School of Electrical Engineering, Tel Aviv University

Abstract

Deep networks usually require a massive amount of labeled data for their training. Yet, such data may include some mistakes in the labels. Interestingly, networks have been shown to be robust to such errors. This work uses spectral analysis of their learned mapping to provide an explanation for their robustness. In particular, we relate the smoothness regularization that usually exists in conventional training to the attenuation of high frequencies, which mainly characterize noise. By using a connection between the smoothness and the spectral norm of the network weights, we suggest that one may further improve robustness via spectral normalization. Empirical experiments validate our claims and show the advantage of this normalization for classification with label noise.

1 INTRODUCTION

Deep neural networks (DNNs) exhibit state-of-the-art results in many machine learning tasks (Goodfellow et al., 2016). Still, their performance heavily relies on the quality of the training data, which - in the supervised scenario - is composed of input-output pairs. In many real-world tasks, the provided outputs, which are commonly referred to as labels, are prone to manual or automatic annotation errors (Liu et al., 2016; Lee et al., 2018), e.g., due to insufficient expertise or to a context-based annotation of web images. Consequently, robustness to such mistakes, known as label noise, is of critical importance for DNNs. Surprisingly, Flatow and Penner (2017); Krause et al. (2016); Sun et al. (2017); Rolnick et al. (2017); Wang et al. (2018) have shown that neural networks exhibit some robustness to this noise. They show that the degradation in network performance can be significantly smaller than the amount of label noise in the training data. While they empirically demonstrate this robustness, our work focuses on analyzing and thus also improving this robustness.

Note though that the robustness of the neural networks to label noise happens only when the noise is not too well structured Drory et al. (2020). We assume that this non-structureness implies that a relatively large part of the noise is in the spectral (Fourier) domain, an assumption that we validate for various noise types and cases (see for example Figure 1). With this assumption, we use the spectral domain to understand the robustness of networks to such types of label noise.

Encouraged by recent advancements in the functional analysis of neural networks (Savarese et al., 2019; Williams et al., 2019; Ongie et al., 2020; Giryes, 2020), we analyze the spectral coefficients of neural networks. This point of view is used to shed light on the relation between the network smoothness and its ability to fit the training data. This tradeoff is controlled by the amount of regularization on the norm of the network derivative with respect to the input. By introducing a bound on this norm, we conclude that the smoothness can increase by imposing constraints on the weights. Following that, we show that further robustness to label noise is obtained by bounding the network weights, as this attenuates high frequencies, which we assume to be mainly stemming from the noise. We validate this assumption in the experiments by using a relationship that we present between the Jacobian and spectral norm of the network to its frequencies.

Our analysis provides a justification for the effectiveness of early stopping and weight decay for DNNs in the presence of label noise. Our theory also suggests using spectral normalization (SN) of the DNN weights (Yoshida and Miyato, 2017; Miyato et al., 2018) as an additional mean for improving robustness to label noise. We show that this simple operation implies a decay in the high frequencies of the mapping learned by the DNN. As ℓ_2-based regularization, SN attends the entire input space at once, rather than only sampled points, and it does not require an addi-

Proceedings of the 25$^{\text{th}}$ International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151.

tional train phase, auxiliary information or extra variables and tuning, which are common in label noise resistance methods. With only a minor computational cost, the trained DNN gains an improved immunity to label noise. To support our theory, we show in various experiments on both synthetic label noise (CIFAR-10, CIFAR-100 (Krizhevsky et al., 2009) and MNIST (Lecun et al., 1998)) and real label noise (Clothing1M (Xiao et al., 2015)) that adding SN to other regularization techniques consistently improves the network performance in the presence of label noise.

Contribution. Our contribution can be summarized by the following three main steps that we perform in it: **a.** Showing that regularizing the network Jacobian either directly or through spectral normalization reduces the high frequency in the learned network mapping. **b.** Demonstrating empirically that having label noise in the training data: (i) adds high frequencies to the learned mapping in the one-dimensional case, where we can practically draw the spectrum of the network; and (ii) increases the Jacobian of the network and its spectral norm in the high-dimensional setting. Both of these steps support our assumption that label noise adds high frequencies to the learned mapping. We borrow this intuition from signal processing, where (random) noise usually lies in all the spectrum compared to the signal that mainly resides in the low frequencies. **c.** Exhibiting that using spectral normalization increases the network robustness to noise. We show that the same holds for Jacobian regularization but more minorly as it does not regularize all the spectrum as spectral normalization does.

2 RELATED WORK

Neural networks resistance to label noise. The natural robustness of neural networks to label noise was empirically investigated in several cases (Krause et al., 2016; Sun et al., 2017; Rolnick et al., 2017; Wang et al., 2018; Drory et al., 2020). Flatow and Penner (2017) showed that while the noise rate increases by tens of percent, test accuracy drops by only a few percent. Nevertheless, the effective dimension of the learned data representation was shown to be different for clean and noisy labels (Ma et al., 2018), suggesting it is possible to further increase the robustness.

Various strategies were proposed to improve the intrinsic resistance of DNNs to label noise. They may be categorized into three groups: (1) probabilistic noise modeling, (2) training data enhancement, and (3) adapted optimization, which may include the objective function, regularization and training procedure. The most common practice for the first is estimating a transition matrix from correct labels to corrupted ones, which is incorporated in the optimization process. The approach in (Patrini et al., 2017) based its matrix estimation on the softmax output of a network trained on a noisy dataset. Alternatively, another route (Goldberger and Ben-Reuven, 2017; Jindal et al., 2016) suggested an end-to-end framework in which the noise distribution is learned simultaneously with the network parameters. Other works leveraged an additional clean data (Xiao et al., 2015; Vahdat, 2107) or manually defined constraints (Han et al., 2018a) to further improve the estimation quality. The second strategy aims at reducing the noise effect by "improving" the provided training dataset, either by rejecting (not using) part of the samples (Shen and Sanghavi, 2019; Han et al., 2018b; Malach and Shalev-Shwartz, 2017), assigning an appropriate weight per sample (Ren et al., 2018; Thulasidasan et al., 2019; Jiang et al., 2018; Liu and Tao, 2015; Guo et al., 2018; Yao et al., 2018) or "correcting" the labels (Reed et al., 2014; Li et al., 2017; Tanaka et al., 2018). The third approach includes the generalized cross-entropy loss (Zhang and Sabuncu, 2018; Amid et al., 2019b,a), symmetric cross-entropy loss (Wang et al., 2019), information-theoretic loss function (Xu et al., 2019a), artificial neural variability Xie et al. (2021), minimum entropy (Reed et al., 2014), mixup (Zhang et al., 2018), and early stopping (Li et al., 2020). The method suggested in this work (SN) falls under this category.

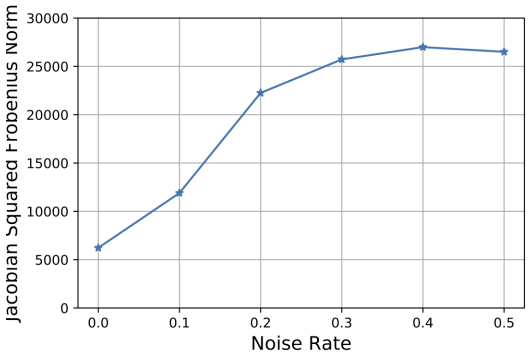
Functional analysis of neural networks. Recently, it was empirically shown that neural networks tend to learn the low frequencies in the data first (Xu et al., 2019b; Bietti and Mairal, 2019; Tancik et al., 2020). In (Rahaman et al., 2019), this behavior was explained by analyzing the Fourier transform of ReLU networks. Another explanation based on approximating the trained network using a linear system was given by (Ronen et al., 2019; Basri et al., 2020). Heckel and Soltanolkotabi (2020) studied the removal of high frequencies by shallow convolutional models that can be used for denoising, e.g., deep image prior (Ulyanov et al., 2017). The behavior of the networks was tied to the convolutional structure. In this work, we propose an alternative explanation for the tendency of networks to prefer learning low frequencies by using the smoothness property of neural networks with bounded weights.

The function space generated by networks with bounded weights was analyzed in various works. Savarese et al. (2019) showed that univariate shallow networks with infinite width and bounded weights represent functions with a bounded total variation of their first derivative. This implied that the learned mapping smoothly interpolates (at least first-order spline) the training points. Ongie et al. (2020) extended this

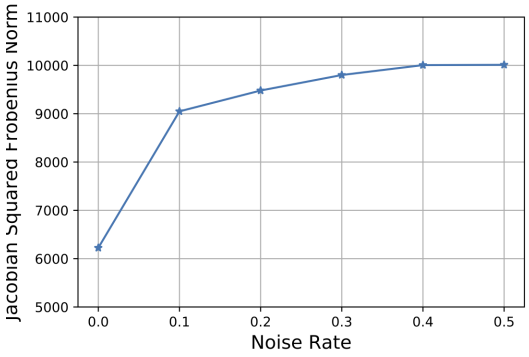
result to the case of shallow networks with multidimensional input. Williams et al. (2019) proved that based on the parameterization of the (univariate) network, one may get a guarantee for second-order spline interpolation of the training data. Giryes (2020) developed generalization bounds for finite networks assuming that the training data was generated by a band-limited mapping. Note that all these works assume that the network overfitted the training data.

We extend these works to the case where no perfect overfitting of the training data is achieved. We show that bounding the weights of the network yields a tradeoff in the loss between fitting the training data and having a smooth mapping. Assuming that mappings of true data are indeed smooth, our theory suggests that the smoothness regularization imposed by bounding the weights provides a "denoising" effect, which helps networks to be resistant to label noise.

Spectral normalization. Yoshida and Miyato (2017) suggested regularizing the spectral norm of neural network weights. Miyato et al. (2018) imposed SN, which directly constrains the spectral norm of each layer and sets it to 1, to stabilize the training of generative adversarial networks. SN was also applied as a means to improve robustness to adversarial attacks (Farnia et al., 2019). In (Neyshabur et al., 2018; Bartlett et al., 2017), network generalization capabilities were analyzed using the weights spectral norm. Our work ties it to label noise robustness both in theory and practice.



(a) Uniform Noise



(b) Flip Noise

Figure 1: Squared Frobenius norm of the Jacobian matrix of a neural network fully fitted to the CIFAR-10 training data, for various noise rates. The Frobenius norm presened is an average over all the data. The behaviour shown here is not unique to CIFAR-10 and is observed also for other datasets.

3 THEORETICAL ANALYSIS

Notation. We use the following notation: scalars, column vectors, matrices and sets are denoted by italic letters (x), boldface lower-case letters ($\mathbf{x}$), boldface upper-case letters ($\mathbf{X}$), and calligraphic upper-case letters ($\mathcal{X}$), respectively. The ith element of a vector $\mathbf{x}$ is denoted by x_i, and the element of the ith row and jth column of $\mathbf{X}$ is denoted by X_{ij}. $\|\mathbf{x}\|_2$, $\|\mathbf{X}\|_2$, and $\|\mathbf{X}\|_F$ denote the Euclidean norm of $\mathbf{x}$, the spectral norm of $\mathbf{X}$, and the Frobenious norm of $\mathbf{X}$, respectively. The all-zeros vector, the all-ones vector, and the identity matrix are denoted by $\mathbf{0}$, $\mathbf{1}$, and $\mathbf{I}$, respectively, with size clear from context. The transpose operation is denoted by ^{T}.

Vector indexing. Given a vector of indices $n = [n_1, \ldots, n_m]$, we abuse notation and use it to index vectors and matrices. This is done by simply converting (uniquely) the tensor indices in n to be vector indices as if the tensor was represented in a column-stack. Thus, we can simply index a vector x using x_n. For a matrix P_{li} (with $l, i \in \mathbb{Z}^m$) denotes indexing the entries of the matrix P after transforming the coordinate in l, i to their corresponding "column-stack" coordinates.

Theoretical analysis outline. We focus on the case of a multivariate neural network ϕ with a single output neuron, L layers, weights $\{\mathbf{W}_l\}_{l=1}^L$ and biases $\{\mathbf{b}_l\}_{l=1}^L$. We consider a bounded input domain $[0, 2\pi]^m$, which is a realistic assumption in real data, where the input range is usually limited (e.g., in images the range is $[0, 1]$ or $[0, 255]$). The network is trained with the pairs $\{(x_n, f(x_n))\}_{x_n \in S}$, where S is the training set and $f : [0, 2\pi]^m \to \mathbb{R}$ is the labels generating function. We assume a uniform sampling scheme of the input domain, i.e., $x_n = \left[\frac{2\pi n_1}{N}, \ldots, \frac{2\pi n_m}{N}\right]$, where $n_i \in \{0, \ldots, N-1\}$ and $n \in \{0, \ldots, N-1\}^m$ (in this case the size of the training set is $|S| = N^m$). Given a vector of indices such as n, we abuse notation and use it to index vectors and matrices. This can be simply done by converting the tensor indices in n to vector indices as done when column-stacking a tensor.

The uniform sampling assumption is used in the proofs of Propositions 1 and 3. It is possible to extend the analysis also to a random sampling scheme, which is usually the case in real data. We comment in each proof on the steps required for this extension. Yet, we defer such a generalization of our study to a future work. Our analysis in the sequel is done in the spectral (Fourier) domain. A short reminder on Fourier properties appears in sup. mat.

We assume ϕ and f are appropriate functions and denote by $\{d_k\}_{k\in\mathbb{Z}^m}$ and $\{c_k\}_{k\in\mathbb{Z}^m}$ their Fourier coefficients, respectively. We use them to analyze the network mapping when optimized with various regularization techniques. We assume that the network attains the global optimum of the optimization in our analysis below (except in our comment on early stopping). This assumption has been shown to be valid in neural network optimization under some assumptions (e.g., see Du et al. (2019)).

We start with the case of regularizing the network derivative (equivalent to regularizing the Jacobian spectral norm in the multidimensional case), and then relate it to penalizing the Frobenius norm (which is referred to as ℓ_2 regularization) and the spectral norm of the network weights. Finally, we suggest using SN and show that it improves robustness to label noise. All proofs are in sup. mat.

3.1 Regularization and Its Effect on The Neural Network Spectrum

Assume we are minimizing the ℓ_2 distance between the outputs of the network and the input labels:

$$\min_{\phi} \frac{1}{|S|} \sum_{x_n \in S} (\phi(x_n) - f(x_n))^2. \tag{1}$$

Note that this loss is relevant for both regression and classification, as it was shown to perform at least as well as cross-entropy in classification tasks (Poggio and Liao, 2020). Clearly, without any constraints on the network and with a sufficient number of parameters in the network, we may bring the empirical error to zero and even overfit the training data. Yet, let us further assume that the mapping we are learning is smooth, i.e., its first derivative is bounded. As we shall see hereafter, neural networks with bounded weights obey this assumption. To this end, we add a regularization on the first derivative of the learned network, i.e., our objective becomes

$$\begin{aligned} \min_{\phi} \quad & \tfrac{1}{|S|} \textstyle\sum_{x_n \in S} (\phi(x_n) - f(x_n))^2 \\ & + \tfrac{\lambda}{(2\pi)^m} \textstyle\int_{x\in[0,2\pi]^m} \left(\tfrac{d\phi(x)}{dx}\right)^2 dx. \end{aligned} \tag{2}$$

Moving to the Fourier domain provides insights about the network bias towards low frequencies.

Proposition 1 *Let $\phi(x) = \sum_{k\in\mathbb{Z}^m} d_k e^{jk^T x}$ be the Fourier series of the trained neural network with uniformly sampled training data. Then, the global optimum of equation 2 obeys*

$$d_k = O\left(\frac{1}{\lambda \left\|k\right\|_2^2}\right),\ k \in \mathbb{Z}^m. \tag{3}$$

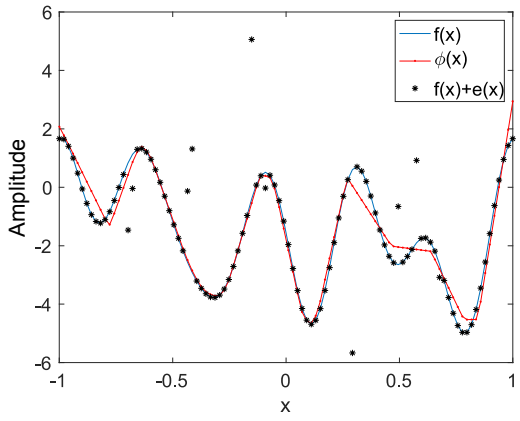
Clearly for $k = 0$, the bound is infinity as the DC component does not alter smoothness. An exact expression for d_k is provided in sup. mat. when the training set size satisfies $N \to \infty$. This proposition may explain the bias of networks with bounded derivative towards low frequencies: the stronger the regularization, the stronger is the decay of the spectral coefficients. As a larger penalty is imposed on the higher frequencies, it is expected that during the training process the network will first learn the lower frequencies. This may explain the usage of early stopping for label noise and stands in line with the observations in (Rahaman et al., 2019; Xu et al., 2019b; Tancik et al., 2020).

While the regularization term in equation 2 applies to the entire input domain, it is clearly impossible to apply such a regularization in practice, as it needs to be evaluated on all possible inputs. The trivial alternative is to apply Jacobian regularization to the training data points (Sokolić et al., 2017; Varga et al., 2017; Jakubovitz and Giryes, 2018; Hoffman et al., 2019). Yet, this translates to only local regularization, which is less effective in practice even when applied randomly, e.g., with mixup (see sup. mat.). Instead, we suggest to upper bound the derivative of the network by its weights:

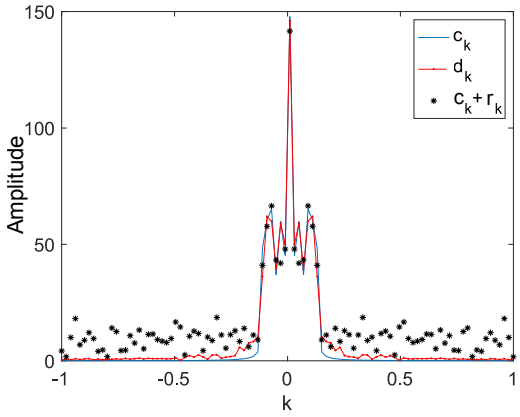
Proposition 2 [based on Lemma 1 by Sokolić et al. (2017)] *Let $\boldsymbol{\phi}(\mathbf{x})$ be a L-layers feed-forward network with a multidimensional input $\mathbf{x} \in \mathcal{X}$, Jacobian matrix $\frac{d\boldsymbol{\phi}(\mathbf{x})}{d\mathbf{x}}$, non-expansive activation functions, and weights and biases $\{\mathbf{W}_l\}_{l=1}^L$ and $\{\mathbf{b}_l\}_{l=1}^L$. Then, we have*

$$\left\|\frac{d\boldsymbol{\phi}(\mathbf{x})}{d\mathbf{x}}\right\|_2^2 \leq \prod_{l=1}^{L} \|\mathbf{W}_l\|_2^2 \leq \prod_{l=1}^{L} \|\mathbf{W}_l\|_F^2, \quad \mathbf{x} \in \mathcal{X}. \tag{4}$$

Note that the assumption on non-expansive activation functions holds for the currently used functions (e.g., ReLU, sigmoid and tanh). The above proposition provides an upper bound for the regularization term in equation 2, which may suggest to replace the regularization on the network derivative with a regularization on the network weights, which is feasible during training. From equation 4, this can be done through a penalty on the weights' spectral or Frobenius norm.



(a) Input domain



(b) Fourier domain

Figure 2: A network $\phi \leftrightarrow d_k$ is fitted to training data generated from $f \leftrightarrow c_k$ with label noise $e \leftrightarrow r_k$ added to 10% of the labels. The original mapping (blue), the network output (red) and the noisy training samples (*) are presented in the input domain (left) and Fourier domain (right). The network "ignores" the high frequencies stemming from the noise.

Notice that using the arithmetic-geometric mean inequality, we may upper bound equation 4 further by $\sum_{l=1}^{L} \|\mathbf{W}_l\|_F^2$, which is the standard ℓ_2 regularization (equivalent to weight decay (WD) in the case that standard SGD optimization is performed (Loshchilov and Hutter, 2019)). Thus, we conclude that training a DNN with WD is expected to regularize the network derivatives and thus lead to a similar effect to the one described in Proposition 1, i.e., fast decay of the high frequency components.

3.2 Network Robustness to Label Noise

We turn to relate our intermediate conclusions to the label noise setting. Consider the case where a noise e is added to the training set, s.t we have $\{(x_n, f(x_n) + e(x_n))\}_{n\in S}$. Now, let us assume a rapid decay of the Fourier coefficients of f, i.e., the function that generates the realizable training set as described above, and that e contains high frequencies compared to the mapping f. Fig. 1 supports this assumption, by exhibiting the existence of high frequencies in noisy (training) data. For this purpose, a training data representation was required. To this end, we train our baseline network until convergence, where the network overfits the noisy data, in the sense that the noisy train accuracy is 100% (early stopping is not applied). High frequencies in the training data are demonstrated by the smoothness of the mapping (measured by the Jacobian Frobenius norm of the fitted network). Fig. 1 depicts the Jacobian measure on CIFAR-10 training data with two types of corrupted labels (see Section 4 for details). Indeed, as the noise level increases, the Jacobian measure increases, indicating that the training data contains higher frequencies.

Since f resides mainly in the low frequencies and e tends to have a lot of high frequencies, from Propositions 1 and 2, we expect that a network trained with WD will learn mainly the low frequencies, i.e., the "clean" data components. This translates to the conclusion that WD introduces a certain level of robustness to label noise. In addition, we conclude that a network trained with WD first learns the low frequencies. Combining this with the assumption that noise mainly resides in high frequencies, we can understand the reason behind the efficiency of early stopping in dealing with noisy labels (when a regularization on the weights is applied). Since the higher frequencies are penalized more, they are likely to be learned later. Thus, early stopping prevents the learning of the high frequencies, and by that filters most of the noise. To summarize, using WD along with early stopping included in conventional training, improves robustness to label noise.

Fig 2 demonstrates this behavior when learning a one-dimensional mapping f, which is composed of a random combination of 6 sine and cosine functions (with a DC component). The data is generated by uniformly sampling 100 points in the range $[-1, 1]$ and then adding random noise to 10% of the samples (randomly selected). We train a fully connected (FC) network with two hidden layers of size 1000 and ReLU, using SGD with momentum, WD, and early stopping. By comparing the Fourier coefficients of the clean (blue) and noisy (*) data, it can be clearly seen that the former resides in the low frequencies, while the high frequencies are mainly occupied by the noise. Notice how the network (red) learns the low frequencies and "ignores" the high frequencies, which aligns with our analysis.

3.3 Network Robustness to Label Noise by Weights Spectral Normalization

Till now we used the unconstrained form of optimization. This led us to insights on how regularizing the weights of the network may improve its robustness to label noise. Now, we turn to analyze the constrained case:

$$\min_{\phi} \frac{1}{|S|} \sum_{x_n \in S} (\phi(x_n) - f(x_n))^2 \quad (5)$$
$$s.t. \quad \frac{1}{(2\pi)^m} \int_{x\in[0,2\pi]^m} \left\| \frac{d\phi(x)}{dx} \right\|_2^2 dx \leq \alpha,$$

where α is a regularization parameter. This allows us to attain direct bounds on the spectral attenuation stemming from bounding the network weights. The next proposition provides the equivalent of Proposition 1 for the constrained case discussed here under an asymptotic regime.

Proposition 3 *Let $\phi(x) = \sum_{k\in\mathbb{Z}^m} d_k e^{jk^T x}$ and $f(x) = \sum_{k\in\mathbb{Z}^m} c_k e^{jk^T x}$ be the Fourier series of the trained neural network and the target mapping function, respectively. If the training set size satisfies $N \to \infty$, then the global optimum of equation 5 is equivalent to the one of*

$$\min_{\{d_k\}_{k\in\mathbb{Z}^m}} \sum_{k\in\mathbb{Z}^m} |d_k - c_k|^2 \quad (6)$$
$$s.t. \quad \sum_{k\in\mathbb{Z}^m} \|k\|_2^2 |d_k|^2 \leq \alpha,$$

and the optimal solution reads as

$$d_k = \begin{cases} c_k & if \ \sum_{k\in\mathbb{Z}^m} \|k\|_2^2 |c_k|^2 < \alpha \\ \frac{c_k}{1+\lambda_\alpha \|k\|_2^2} & otherwise \end{cases}, \quad (7)$$

where λ_α is the solution to the equation $\sum_{k\in\mathbb{Z}^m} \|k\|_2^2 \frac{|c_k|^2}{(1+\lambda_\alpha\|k\|_2^2)^2} = \alpha$.

This proposition shows that also in the constrained case, if the constraint is non-trivial (i.e., does not affect the solution), then higher frequencies are more penalized. To draw a relationship between constraining the network weights and attenuating the high frequencies, we consider the following optimization problem that constraints the spectral norm of the network weights:

$$\min_{\phi} \frac{1}{|S|} \sum_{x_n \in S} (\phi(x_n) - f(x_n))^2 \quad (8)$$
$$s.t. \quad \prod_{l=1}^{L} \|\mathbf{W}_l\|_2^2 \leq \alpha.$$

Notice that from Proposition 2, we have that the feasible set in equation 8 is included in the one of equation 5, i.e., if $\prod_l \|\mathbf{W}_l\|_2^2 \leq \alpha$ then also $\frac{1}{(2\pi)^m} \int_{x\in[0,2\pi]^m} \left\| \frac{d\phi(x)}{dx} \right\|_2^2 dx \leq \alpha$ holds. Therefore, it is expected that applying a constraint on the network weight matrices norms will attenuate the high frequencies in the learned mapping. Clearly, we could have used also the Frobenius norm according to Proposition 2 as a bound. Yet, since it is a weaker upper bound (as shown in Proposition 2), we expect that using it as a proxy will lead to inferior results compared to the case of using the spectral norm. As shown in Section 4, this is indeed the case when training neural networks with label noise. Applying the constraint in equation 8 directly is computationally hard. Thus, instead of bounding the product of the layers weights norms we suggest bounding each norm separately: $\|\mathbf{W}_l\|_2 \leq \alpha_l,\ l = 1, \ldots, L$. Notice that if the constraints are not trivial, they become $\|\mathbf{W}_l\|_2 = \alpha_l,\ l = 1, \ldots, L$. For the case $\alpha_l = 1,\ l = 1, \ldots, L$, this regularization is known as SN (Miyato et al., 2018). Combining it with an arbitrary loss function ℓ we have

$$\min_{\phi} \frac{1}{|S|} \sum_{x_n \in S} \ell\left(\phi(x_n), f(x_n)\right) \quad (9)$$
$$s.t. \quad \|\mathbf{W}_l\|_2 = 1,\ l = 1, \ldots, L.$$

The next proposition shows that SN encourages a decay of the learned map spectral coefficients.

Proposition 4 *Let $\phi(x) = \sum_{k\in\mathbb{Z}^m} d_k e^{jk^T x}$ be the Fourier series of the trained neural network. Then, the global optimum of equation 9 obeys*

$$|d_k| \leq \frac{1}{\|k\|_2},\ k \in \mathbb{Z}^m. \quad (10)$$

This proposition shows that SN encourages learning a mapping with decaying spectral coefficients. Thus, it is expected to improve network robustness to label noise. It is possible to extend this result to normalization to a constant other than 1. In this case, we get the same bound as before but with $\alpha = \prod_{l=1}^{L} \alpha_l$ in the nominator of the right-hand-size of equation 10.

To appreciate why using the spectral norm can be a good approximation to a regularization on the Jacobian of the network in all locations consider the following simple case of two layer linear network $\phi(x) = W_2(W_1 x + b_1)$. In this case, the norm of the Jacobian of the network is $\|W_2 W_1\|_2$, which is the spectral norm of $W_2 W_1$. In our case, we regularize each of them independently using the bound $\|W_2 W_1\|_2 \leq \|W_2\|_2 \|W_1\|_2$, which is tight since we get equality in it when the right singular vectors of W_1 are equal to the left singular vectors of W_2.

While the decay rate in Proposition 4 is weaker than the one in Proposition 3, the actual decay rates might

be stronger. Moreover, the guarantee in Proposition 4 is independent of the training size and the loss function used, i.e., it applies also to minimization with the categorical cross-entropy loss in the case of classification. Note also that for the proposition we do not need the assumption of uniform sampling assumption. As we show next, this regularization indeed improves the label noise robustness in real data classification.

4 EXPERIMENTS

In this section, our theoretical findings are validated in the framework of image classification with label noise. We empirically examine the application of SN in synthesized noisy datasets, with a variety of noise patterns and rates, as well as in a real-world noisy dataset. Our experiments demonstrate that indeed, using SN improves performance over conventional training, independently of both architecture and dataset. Moreover, SN improvement is evident even when combined with other methods. Experiments technical details are provided in sup. mat. The code for performing the experiments is available in https://github.com/OshratBar/A-Spectral-Perspective-of-Neural-Networks-Robustness-to-Label-Noise.

We added synthetic label noise of two popular statistical models (uniform and flip noise; see Patrini et al. (2017)) in various rates to CIFAR-10, CIFAR-100, and MNIST. All convolutional network (Springenberg et al., 2014) and LeNet-5 (LeCun et al., 1998) were utilized to classify the CIFAR datasets and MNIST, respectively. For the baseline networks, we used cross-entropy loss with ℓ_2 regularization, and applied early stopping (according to the validation set accuracy). All details regarding the datasets, artificial noise, networks, training procedures and hyperparameters are specified in sup. mat. We present here results for CIFAR-10; The results for CIFAR-100 and MNIST are reported in sup. mat. In addition, comparison with Jacobian regularization is presented in sup. mat., and confirms that SN is better.

4.1 Bounding the Network Weights Increases its Smoothness

Before showing the effect of weight regularization on accuracy, we first validate a core claim in our analysis: regularizing the network weights either by ℓ_2 or by SN improves its smoothness. To do so, we calculate the averaged squared Frobenius norm of the network Jacobian over the test data, which is a measure of the network smoothness. We check several configurations: without regularization (except early stopping), with the baseline ℓ_2, 100 times increased ℓ_2 (strong ℓ_2), and ℓ_2 with SN. Table 1 displays the Jacobian measure for CIFAR-10 in all noise rate-regularization combinations for uniform and flip noises. It can be seen that as the level of ℓ_2 regularization rises, the DNN is smoother, while the ℓ_2 with SN configuration provides the smoothest result. As we show next, this is done without compromising the classification accuracy, but vice-versa.

Notice the difference between the Frobenius norms in Table 1 and Figure 1. We see a different results as early stopping (regularization) is applied in Table 1 while in Figure 1 the training data including the noisy examples are fully overfitted. Notice that as the noise level increases, the early stopping regularization has a stronger effect, and the yielded network is smoother. This can be explained by the observation that early stopping forces the network to learn mainly from the clean data. Therefore, as the noise level raises, the effective train data is smaller and therefore it is easier to explain it by a simpler (smoother) function.

4.2 Classification With Spectral Normalization

We turn to demonstrate the contribution of adding SN in various classification tasks with label noise. Tables 2 and 3 present the test accuracy for CIFAR-10 dataset, corrupted by uniform and flip noise, respectively. The regularizations used are as in Table 1. We also present SN alone for the uniform case. Note that it improves over the baseline and also ℓ_2 but it can be seen that in all noise levels, ℓ_2 regularization combined with SN gains the highest test accuracy. Therefore, we use them together. Note that both of them bound the network derivatives and thus regularize its smoothness according to our analysis in Section 3. Secondly, our expectation that higher smoothness of the network increases the resistance to label noise is correct for all subjected cases except for strong ℓ_2. Compared to baseline ℓ_2, strong ℓ_2 squeezes the Jacobian, but degrades the test accuracy. This may happen as a high ℓ_2 coefficient overshadows the cross-entropy loss weight. In contrast, SN imposes smoothness without affecting the learning process.

To show that the accuracy improvement is indeed stemming from SN, we present the spectral norms of the layers of the baseline network in Table 4. The first thing that the table shows is that indeed the spectral norm of the network layers increase as the noise rate becomes larger. This justifies the assumption we make in Section 3 indicating that the noise adds high frequency components in the network mapping and makes it less smooth. Notice that SN reduces the spectral norms of the network (as they are greater than 1 in the baseline) and thus improve performance. All

Table 1: *Bounding the network weights increases its smoothness.* Squared Frobenius norm of the neural network Jacobian matrix, averaged over CIFAR-10 test data, for various noise rates and regularization methods. Notice that early stopping is applied here, unlike 1 where there is an overfitting of the training data.

Regularization \ Noise Rate	0	Uniform Noise 0.1	0.3	0.5	0.7	Flip Noise 0.1	0.2	0.3	0.4	0.5
No	4736	2411	1055	329	100	3125	2789	2608	2484	2426
ℓ_2	4500	2312	942	357	90	3094	2654	2620	2362	3181
Strong ℓ_2	1785	1060	378	442	**23**	1537	1400	1258	1128	1199
ℓ_2 **+ SN**	**465**	**434**	**362**	**233**	79	**377**	**343**	**324**	**349**	**358**

Table 2: CIFAR-10 test accuracy when trained with different rates of uniform noise and different regularization methods.

Regularization \ Noise Rate	0	0.1	0.3	0.5	0.7
No	89.61±0.11	86.87±0.21	82.95±0.36	78.53±0.39	69.80±0.20
ℓ_2	90.03±0.19	87.42±0.10	83.57±0.52	79.28±0.37	**69.88**±0.81
Strong ℓ_2	80.63±0.45	76.84±0.80	64.75±1.23	71.34±0.31	45.10±3.28
SN	90.60±0.07	89.18±0.13	84.93±0.16	79.88±0.26	69.55±0.38
ℓ_2 **+ SN**	**90.82**±0.21	**89.32**±0.22	**85.35**±0.25	**80.22**±0.16	69.79±0.45

Table 3: CIFAR-10 test accuracy when trained with different rates of flip noise and different regularization methods.

Regularization \ Noise Rate	0.1	0.2	0.3	0.4	0.5
No	87.63±0.30	85.93±0.24	83.39±0.30	80.76±0.43	72.45±0.72
ℓ_2	88.41±0.40	87.55±0.23	85.81±0.52	82.47±0.38	73.19±0.98
Strong ℓ_2	78.43±0.23	76.64±0.47	72.53±0.50	68.59±0.54	61.29±0.42
ℓ_2 **+ SN**	**89.69**±0.19	**88.22**±0.41	**86.03**±0.40	**82.97**±0.49	**73.24**±0.20

Table 4: Spectral norms of the baseline network layers.

Noise rate \ Layer	1	2	3	4	5	6	7	8	9
0	3.60	5.01	4.33	5.94	5.89	5.93	4.99	5.89	2.04
0.1	3.46	4.96	4.35	5.87	5.94	5.65	5.00	6.91	1.99
0.3	3.46	4.96	4.35	5.87	5.94	5.65	5.00	6.91	1.99
0.5	4.24	8.68	7.37	9.90	11.11	11.47	11.04	13.96	2.48
0.7	4.09	8.17	7.03	8.36	8.35	7.87	9.00	9.25	2.47

Table 5: *SN combined with mixup.* CIFAR-10 test accuracy when trained with different rates of uniform noise, using SN and mixup.

Regularization \ Noise Rate	0	0.1	0.3	0.5	0.7
ℓ_2	90.03±0.19	87.42±0.10	83.57±0.52	79.28±0.37	69.88±0.81
ℓ_2 **+ SN**	90.82±0.21	89.32±0.22	85.35±0.25	80.22±0.16	69.79±0.45
ℓ_2 **+ Mixup**	90.35±0.17	88.03±0.18	84.63±0.28	79.41±0.40	71.14±0.43
ℓ_2 **+ Mixup + SN**	**91.47**±0.10	**89.94**±0.15	**86.04**±0.09	**80.29**±0.23	**71.43**±0.35

this stand in line with our claims above that relate smoothness to robustness.

4.3 SN Combined With Other Regularization Methods

To emphasize the fact that SN can be combined with other label noise resistance methods (in addition to early stopping) and still gain a performance improvement, we also report its performance when used along with mixup and minimum entropy regularization (Grandvalet and Bengio, 2005, 2006; Lee, 2013). Mixup is a simple and data-agnostic data augmentation routine, which extends the training distribution via convex combinations of training examples pairs. Mixup encourages the model to behave linearly in-between training examples, and by that implicitly controls model complexity. Furthermore, it was empirically found that mixup reduces the memorization of corrupted labels. Table 5 reports test accuracy of CIFAR-10 with different rates of uniform noise, when regularized by SN, mixup, and their combination. Mixup improves the baseline accuracy, and the addition of SN increases it even more. Results for SN combined with minimum entropy regularization are reported in sup. mat. The behaviour of SN improvement is observed also there.

4.4 Real Noise

Clothing1M dataset contains 1M images of clothing obtained from online shopping websites. The images are classified to one of 14 classes by using their surrounding texts, and therefore the labels contain many errors. A small portion of the noisy labels was manually refined and split into training, validation and test sets of sizes 50K, 14K and 10K, respectively. As commonly done for this dataset, e.g., in (Patrini et al., 2017; Wang et al., 2019; Xu et al., 2019a), we also used a bottleneck ResNet-50 (He et al., 2016) pre-trained on ImageNet (Deng et al., 2009). For preprocessing, we used a resize to 256×256, middle crop to 224×224 and mean-subtraction as in Tanaka et al. (2018). As opposed to most works, no data augmentation was performed, and our fine-tuning did not utilize the additionally provided clean training set. We used SGD with momentum 0.9, a batch size of 32, and an ℓ_2 regularization coefficient of 10^{-3}, for two epochs, with learning rates $8 \cdot 10^{-4}$ and $8 \cdot 10^{-5}$. As Table 6 shows, adding SN improves test accuracy, when applied both with and without mixup.

Table 6: Clothing1M test accuracy

ℓ_2	ℓ_2 + SN	ℓ_2 + mixup	ℓ_2 + mixup + SN
69.12	70.01	70.3	70.59

5 CONCLUSION

In this paper, the natural robustness of DNNs to label noise is investigated from a new point of view. A spectral-domain analysis is used to provide theoretical tradeoffs between the data fitting and the interpolation smoothness. We show that this trade-off can be controlled by the level of the network weights regularization. We use these findings to get new insights with respect to networks robustness to label noise. By leveraging the observation that label noise imposes high frequencies in the training data, it is concluded that bounding the network weights increases its robustness. Consequently, we justify the commonly used ℓ_2 and early stopping regularizations in the presence of label noise. Furthermore, we suggest using SN, which constitutes a tighter bound on the network derivatives (compared to ℓ_2) and attends the entire input space at once. In addition, since the suggested method has distinct and complementary properties for the subjected problem, it can be integrated into other strategies, to further improve the resistance to label noise. Numerical results confirm the validity of our theoretical findings and proposed strategy on various datasets.

The theory in our work holds for any non-expansive activation function. Such activation functions include ReLU, Tanh, Sigmoid, Leaky ReLU, etc. Our analysis is general and does not depend on the network width or depth (that indeed impact the optimization of the network, which we assume that is done by some optimizer). Moreover, we believe our results can be applied also to other learning tools such as SVM.

Our analysis of the neural network robustness assume that the noise do not reside only in the low frequencies but also has a relatively good portion in the high frequencies, which we show to be filtered during the network training due to the regularizations imposed. Thus, our analysis holds for any type of noise that obey these assumptions. Note though that for some noise types, e.g. structured noise, which typically has few high frequencies, neural networks are not robust and their performance degrade a lot Drory et al. (2020). In this work, we used two types of noise, namely random uniform noise and flip noise, which obey our assumptions. Moreover, We also demonstrated our regularization on real world noise (Fashion1M).

Acknowledgements

This research was supported by the ERC-StG SPADE grant no. 757497

References

Amid, E., Warmuth, M. K., Anil, R., and Koren, T. (2019a). Robust bi-tempered logistic loss based on bregman divergences. In *Advances in Neural Information Processing Systems*, pages 15013–15022.

Amid, E., Warmuth, M. K., and Srinivasan, S. (2019b). Two-temperature logistic regression based on the tsallis divergence. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2388–2396. PMLR.

Bartlett, P. L., Foster, D. J., and Telgarsky, M. (2017). Spectrally-normalized margin bounds for neural networks. In *NeurIPS*.

Basri, R., Galun, M., Geifman, A., Jacobs, D., Kasten, Y., and Kritchman, S. (2020). Frequency bias in neural networks for input of non-uniform density. *arXiv preprint arXiv:2003.04560*.

Bietti, A. and Mairal, J. (2019). On the inductive bias of neural tangent kernels. In *Advances in Neural Information Processing Systems*, volume 32.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR*.

Drory, A., Avidan, S., and Giryes, R. (2020). The resistance to label noise in k-nn and dnn depends on its concentration. In *British Machine Vision Virtual Conference (BMVC)*.

Du, S., Lee, J., Li, H., Wang, L., and Zhai, X. (2019). Gradient descent finds global minima of deep neural networks. In *ICML*.

Farnia, F., Zhang, J., and Tse, D. (2019). Generalizable adversarial training via spectral normalization. In *ICLR*.

Flatow, D. and Penner, D. (2017). On the robustness of convnets to training on noisy labels. Technical report, Stanford University.

Giryes, R. (2020). A function space analysis of finite neural networks with insights from sampling theory. *arXiv preprint arXiv:2004.06989*.

Goldberger, J. and Ben-Reuven, E. (2017). Training deep neural-networks using a noise adaptation layer. In *ICLR*.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

Grandvalet, Y. and Bengio, Y. (2005). Semi-supervised learning by entropy minimization. In *NeurIPS*.

Grandvalet, Y. and Bengio, Y. (2006). Entropy regularization. In *Semi-supervised learning*, pages 151–168. MIT Press Boston, MA, USA.

Guo, S., Huang, W., Zhang, H., Zhuang, C., Dong, D., Scott, M. R., and Huang, D. (2018). Curriculumnet: Weakly supervised learning from large-scale web images. In *ECCV*.

Han, B., Yao, J., Niu, G., Zhou, M., Tsang, I., Zhang, Y., and Sugiyama, M. (2018a). Masking: A new perspective of noisy supervision. In *NeurIPS*.

Han, B., Yao, Q., Yu, X., Niu, G., Xu, M., Hu, W., Tsang, I., and Sugiyama, M. (2018b). Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *NeurIPS*.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *CVPR*.

Heckel, R. and Soltanolkotabi, M. (2020). Denoising and regularization via exploiting the structural bias of convolutional generators. In *ICLR*.

Hoffman, J., Roberts, D. A., and Yaida, S. (2019). Robust learning with jacobian regularization. *arXiv preprint arXiv:1908.02729*.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*.

Jakubovitz, D. and Giryes, R. (2018). Improving DNN robustness to adversarial attacks using Jacobian regularization. In *ECCV*.

Jakubovitz, D., Giryes, R., and Rodrigues, M. R. D. (2019). Generalization error in deep learning. In *Compressed Sensing and Its Applications*, pages 153–193. Springer International Publishing.

Jiang, L., Zhou, Z., Leung, T., Li, L.-J., and Fei-Fei, L. (2018). MentorNet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *ICML*.

Jindal, I., Nokleby, M., and Chen, X. (2016). Learning deep networks from noisy labels with dropout regularization. In *ICDM*.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Krause, J., Sapp, B., Howard, A., Zhou, H., Toshev, A., Duerig, T., Philbin, J., and Fei-Fei, L. (2016). The unreasonable effectiveness of noisy data for fine-grained recognition. In *ECCV*.

Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images.

Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. In *Proc. IEEE*, pages 2278–2324.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324.

Lee, D.-H. (2013). Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *ICML workshop on challenges in representation learning*.

Lee, K.-H., He, X., Zhang, L., and Yang, L. (2018). CleanNet: Transfer learning for scalable image classifier training with label noise. In *CVPR*.

Li, M., Soltanolkotabi, M., and Oymak, S. (2020). Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. In *AISTATS*.

Li, Y., Yang, J., Song, Y., Cao, L., Luo, J., and Li, L.-J. (2017). Learning from noisy labels with distillation. In *ICCV*.

Liu, T. and Tao, D. (2015). Classification with noisy labels by importance reweighting. *IEEE trans. pattern anal. mach. intell.*, 38(3):447–461.

Liu, Z., Luo, P., Qiu, S., Wang, X., and Tang, X. (2016). Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *CVPR*.

Loshchilov, I. and Hutter, F. (2019). Decoupled weight decay regularization. In *ICLR*.

Ma, X., Wang, Y., Houle, M. E., Zhou, S., Erfani, S., Xia, S., Wijewickrema, S., and Bailey, J. (2018). Dimensionality-driven learning with noisy labels. In *ICML*.

Malach, E. and Shalev-Shwartz, S. (2017). Decoupling "when to update" from "how to update". In *NeurIPS*.

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. (2018). Spectral normalization for generative adversarial networks. In *ICLR*.

Neyshabur, B., Bhojanapalli, S., and Srebro, N. (2018). A pac-bayesian approach to spectrally-normalized margin bounds for neural networks. In *ICLR*.

Ongie, G., Willett, R., Soudry, D., and Srebro, N. (2020). A function space view of bounded norm infinite width ReLU nets: The multivariate case. In *ICLR*.

Oppenheim, A. V., Willsky, A. S., and Nawab, S. H. (1997). *Signals and Systems*. Prentice-Hall, Inc., USA.

Patrini, G., Rozza, A., Krishna Menon, A., Nock, R., and Qu, L. (2017). Making deep neural networks robust to label noise: A loss correction approach. In *CVPR*.

Poggio, T. and Liao, Q. (2020). Implicit dynamic regularization in deep networks. Technical report, Center for Brains, Minds and Machines (CBMM).

Rahaman, N., Baratin, A., Arpit, D., Draxler, F., Lin, M., Hamprecht, F., Bengio, Y., and Courville, A. (2019). On the spectral bias of neural networks. In *ICML*.

Reed, S., Lee, H., Anguelov, D., Szegedy, C., Erhan, D., and Rabinovich, A. (2014). Training deep neural networks on noisy labels with bootstrapping. *arXiv preprint arXiv:1412.6596*.

Ren, M., Zeng, W., Yang, B., and Urtasun, R. (2018). Learning to reweight examples for robust deep learning. In *ICML*.

Rolnick, D., Veit, A., Belongie, S., and Shavit, N. (2017). Deep learning is robust to massive label noise. *arXiv preprint arXiv:1705.10694*.

Ronen, B., Jacobs, D., Kasten, Y., and Kritchman, S. (2019). The convergence rate of neural networks for learned functions of different frequencies. In *NeurIPS*.

Savarese, P., Evron, I., Soudry, D., and Srebro, N. (2019). How do infinite width bounded norm networks look in function space? In *COLT*.

Shen, Y. and Sanghavi, S. (2019). Learning with bad training data via iterative trimmed loss minimization. In *ICML*.

Sokolić, J., Giryes, R., Sapiro, G., and Rodrigues, M. R. (2017). Robust large margin deep neural networks. *IEEE Transactions on Signal Processing*, 65(16):4265–4280.

Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. (2014). Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*.

Sun, C., Shrivastava, A., Singh, S., and Gupta, A. (2017). Revisiting unreasonable effectiveness of data in deep learning era. In *ICCV*.

Tanaka, D., Ikami, D., Yamasaki, T., and Aizawa, K. (2018). Joint optimization framework for learning with noisy labels. In *CVPR*.

Tancik, M., Srinivasan, P. P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J. T., and Ng, R. (2020). Fourier features let networks learn high frequency functions in low dimensional domains. *arXiv preprint arXiv:2006.10739*.

Thulasidasan, S., Bhattacharya, T., Bilmes, J. A., Chennupati, G., and Mohd-Yusof, J. (2019). Com-

bating label noise in deep learning using abstention. In *ICML*.

Ulyanov, D., Vedaldi, A., and Lempitsky, V. S. (2017). Deep image prior. In *CVPR*.

Vahdat, A. (2107). Toward robustness against label noise in training deep discriminative neural networks. In *NeurIPS*.

Varga, D., Csiszárik, A., and Zombori, Z. (2017). Gradient regularization improves accuracy of discriminative models. *arXiv preprint arXiv:1712.09936*.

Wang, F., Chen, L., Li, C., Huang, S., Chen, Y., Qian, C., and Loy, C. C. (2018). The devil of face recognition is in the noise. In *ECCV*.

Wang, Y., Ma, X., Chen, Z., Luo, Y., Yi, J., and Bailey, J. (2019). Symmetric cross entropy for robust learning with noisy labels. In *ICCV*.

Williams, F., Trager, M., Panozzo, D., Silva, C., Zorin, D., and Bruna, J. (2019). Gradient dynamics of shallow univariate relu networks. In *NeurIPS*.

Xiao, T., Xia, T., Yang, Y., Huang, C., and Wang, X. (2015). Learning from massive noisy labeled data for image classification. In *CVPR*.

Xie, Z., He, F., Fu, S., Sato, I., Tao, D., and Sugiyama, M. (2021). Artificial neural variability for deep learning: On overfitting, noise memorization, and catastrophic forgetting. *Neural Computation*, 33(8):2163–2192.

Xu, Y., Cao, P., Kong, Y., and Wang, Y. (2019a). L_{DMI}: A novel information-theoretic loss function for training deep nets robust to label noise. In *NeurIPS*.

Xu, Z.-Q. J., Zhang, Y., and Xiao, Y. (2019b). Training behavior of deep neural network in frequency domain. In *ICONIP*.

Yao, J., Wang, J., Tsang, I. W., Zhang, Y., Sun, J., Zhang, C., and Zhang, R. (2018). Deep learning from noisy image labels with quality embedding. *IEEE Trans. Image Process.*, 28(4):1909–1922.

Yoshida, Y. and Miyato, T. (2017). Spectral norm regularization for improving the generalizability of deep learning. *arXiv preprint arXiv:1705.10941*.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. (2018). mixup: Beyond empirical risk minimization. In *ICLR*.

Zhang, Z. and Sabuncu, M. R. (2018). Generalized cross entropy loss for training deep neural networks with noisy labels. In *NeurIPS*.

Supplementary Material: A Spectral Perspective of DNN Robustness to Label Noise

A CLASSIFICATION WITH JACOBIAN REGULARIZATION

During the work process, the direct regularization of the network derivatives was empirically examined, in addition to the indirect regularization through the network weights. Here we present the effect of the Jacobian regularization (Sokolić et al., 2017; Varga et al., 2017; Jakubovitz and Giryes, 2018; Hoffman et al., 2019) on smoothness and accuracy, compared with SN. Table 7 reports the averaged squared Frobenius norm of the network Jacobian matrix, over CIFAR-10 test data, for various rates of uniform and flip noises in the train data. Almost in all cases, the addition of SN to the baseline network is more effective than the addition of Jacobian regularization for smoothing out the network. It is interesting to note that for extremely high rates of uniform noise, Jacobian regularization is more effective. Tables 8 and 9 present CIFAR-10 test accuracy for uniform and flip noises, respectively. With correspondence to the smoothness results, SN increases accuracy more than Jacobian in all cases, except for the extremely high noise rates. This correspondence between the smoothness and the achieved accuracy coincides with our expectation that higher smoothness of the network increases the resistance to label noise. Note that we also applied Jacobian regularization along with mixup (see Table 8). This flavor of Jacobian regularization is broader than the vanilla form, as it attends many input-domain points, rather than only the given training points. While this improves accuracy in most cases compared with vanilla Jacobian, SN performance is still superior.

To summarize, even though Jacobian regularization applies directly to the network derivatives, it is not always the best option, and regularization through the weights can be better. This conforms with our claim that Jacobian regularization is limited by the fact that it attends only to sampled points. Thus, we choose to focus our effort on the weight-based regularizations and leave the Jacobian regularization for future work. This may include understanding the relations between the regularization methods, and leveraging it to compose an optimal combination of them.

B SPECTRAL ANALYSIS REMINDER

As a reminder, we present here the relevant basics of spectral analysis. A function $g : [0, 2\pi]^m \to \mathbb{C}$ is considered appropriate if the following holds (Oppenheim et al., 1997):

1. g satisfies Dirichlet condition.
2. g is squared integrable.
3. The Jacobian g' exists and has a finite number of discontinuities.
4. In the one dimensional case $g(0) = g(2\pi)$, $g'(0) = g'(2\pi)$. In the multidimensional case, the same holds when adding π to any of the coordinates.

Such an appropriate function satisfies the following properties:

- Fourier series:

$$g(x) = \sum_{k \in \mathbb{Z}^m} \alpha_k e^{jk^T x}, \tag{11}$$
$$\alpha_k = \frac{1}{(2\pi)^m} \int_{x \in [0,2\pi]^m} g(x) e^{-jk^T x} dx, \; k \in \mathbb{Z}^m.$$

Table 7: *Jacobian regularization - network smoothness.* Squared Frobenius norm of the network Jacobian matrix, averaged over CIFAR-10 test data for various noise rates, and when trained with SN or Jacobian regularization.

Regularization \ Noise Rate	0	Uniform Noise 0.1	0.3	0.5	0.7	Flip Noise 0.1	0.2	0.3	0.4	0.5
ℓ_2 + **SN**	**465**	**434**	**362**	233	79	**377**	**343**	**324**	**349**	**358**
ℓ_2 + **Jacob**	2390	686	392	**90**	**19**	2145	1962	1904	705	547

Table 8: *Jacobian regularization - network accuracy.* CIFAR-10 test accuracy when trained with different rates of uniform noise and with SN or Jacobian regularization.

Regularization \ Noise Rate	0	0.1	0.3	0.5	0.7
ℓ_2 + **SN**	**90.82**±0.21	**89.32**±0.22	**85.35**±0.25	**80.22**±0.16	69.79±0.45
ℓ_2 + **Jacob**	89.87±0.59	87.66±0.30	84.10±0.47	79.73±0.46	**72.22**±0.30
ℓ_2 + **Jacob + mixup**	90.57±0.19	87.96±0.24	84.89±0.31	79.72±0.34	71.26±0.55

- Derivative property:

$$\frac{dg(x)}{dx_i} = \sum_{k\in\mathbb{Z}^m} jk_i\alpha_k e^{jk^T x}. \tag{12}$$

- Parseval's theorem:

$$\sum_{k\in\mathbb{Z}^m} \|\alpha_k\|_2^2 = \frac{1}{(2\pi)^m}\int_{x\in[0,2\pi]^m} |g(x)|^2 dx. \tag{13}$$

- Norm of Jacobian property (combination of the derivative property and Parseval's theorem)

$$\frac{1}{(2\pi)^m}\int_{x\in[0,2\pi]^m} \left\|\frac{dg(x)}{dx}\right\|_2^2 dx = \sum_{k\in\mathbb{Z}^m} \|k\|_2^2 \|\alpha_k\|_2^2 . \tag{14}$$

C PROPOSITIONS PROOFS

C.1 Proof of Proposition 1

Proof. We use the Fourier series with uniform sampling, i.e., $x_n = \left[\frac{2\pi n_1}{N}, \dots, \frac{2\pi n_m}{N}\right]$, where $n_i \in \{0, \dots, N-1\}$ and $n \in \{0, \dots, N-1\}^m$. Therefore, we have that

$$\begin{aligned}
&\frac{1}{N^m}\sum_{n\in\{0,\dots,N-1\}^m} (\phi(x_n) - f(x_n))^2 \qquad (15)\\
&= \frac{1}{N^m}\sum_{n\in\{0,\dots,N-1\}^m} \\
&\qquad \left(\sum_{k\in\mathbb{Z}^m} (d_k - c_k)e^{jk^T x_n}\right)^* \left(\sum_{q\in\mathbb{Z}^m} (d_q - c_q)e^{jq^T x_n}\right)\\
&= \sum_{k\in\mathbb{Z}^m}\sum_{q\in\mathbb{Z}^m} (d_k - c_k)^* (d_q - c_q) \\
&\qquad \cdot \frac{1}{N^m}\sum_{n\in\{0,\dots,N-1\}^m} e^{j\frac{2\pi}{N}(q-k)^T n}\\
&= \sum_{k\in\mathbb{Z}^m}\sum_{l\in\mathbb{Z}^m} (d_k - c_k)^* (d_{k+lN} - c_{k+lN}) .
\end{aligned}$$

Table 9: *Jacobian regularization - network accuracy.* CIFAR-10 test accuracy when trained with different rates of flip noise and with SN or Jacobian regularization.

Regularization \ Noise Rate	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**
ℓ_2 + **SN**	**89.69**±0.19	**88.22**±0.41	**86.03**±0.40	**82.97**±0.49	73.24±0.20
ℓ_2 + **Jacob**	88.41±0.16	87.50±0.16	85.69±0.25	82.43±0.36	**75.39**±1.02

The last equality is due to the fact that $\frac{1}{N^m}\sum_{n\in\{0,\ldots,N-1\}^m} e^{j\frac{2\pi}{N}(q-k)^T n}$ is 1 if $q = k + lN, l \in \mathbb{Z}^m$. Otherwise, it becomes a geometric series sum with a common ratio of $e^{j(q-k)\frac{2\pi}{N}}$, which equals 0. This can be also seen by noticing that the sum is basically an inner product between the column-stacked columns of a multidimensional DFT. Using Parseval's theorem and the derivative property of the Fourier coefficients for the regularization term, we can rewrite equation 2 as:

$$\begin{aligned}\min_{\{d_k\}_{k\in\mathbb{Z}^m}} \sum_{k\in\mathbb{Z}^m}\sum_{l\in\mathbb{Z}^m}(d_k - c_k)^*(d_{k+lN} - c_{k+lN}) \\ +\lambda\sum_{k\in\mathbb{Z}^m}\|k\|_2^2 |d_k|^2 .\end{aligned} \tag{16}$$

Taking the derivative w.r.t d_k and equating to zero, we have

$$\sum_{l\in\mathbb{Z}^m}(d_{k+lN} - c_{k+lN}) + \lambda\|k\|_2^2 d_k = 0, \quad k \in \mathbb{Z}^m. \tag{17}$$

Rearranging yields

$$\lambda\|k\|_2^2 d_k + \sum_{l\in\mathbb{Z}^m} d_{k+lN} = \sum_{l\in\mathbb{Z}^m} c_{k+lN}, \quad k \in \mathbb{Z}^m. \tag{18}$$

equation 18 represents an infinite system of equations for all the spectral coefficients $\{d_k\}_{k\in\mathbb{Z}^m}$. Note that each Fourier coefficient d_k depends on the coefficients of ϕ and f, whose index distance from k is a multiple of N (in m possible dimensions). Accordingly, we can partition the indices vectors to N^m sets, such that each set is represented by a vector $n \in \{0,\ldots,N-1\}^m$ such that all of its vectors result with the reminder n when dividing their elements by N. The indexes in each set are uniformly spaced and have a gap of N from each other in each dimension. Following that, we can split equation 18 to N^m systems, each corresponding to one of the N^m index sets. Then, for a given set, denote

- $c \triangleq \sum_{l\in\mathbb{Z}^m} c_{k+lN} \in \mathbb{C}^m$ is the sum of f coefficients with indexes belonging to the set
- $\mathbf{u}$ is an infinite sequence (represented as an "infinite vector") of the coefficients of ϕ with indexes belonging to the set, i.e., $u_l = d_{k+lN}, l \in \mathbb{Z}^m$
- $\mathbf{1}$ is an "infinite vector" with all ones, i.e, $1_l = 1, l \in \mathbb{Z}^m$. Note that we could just have used $l \in \mathbb{Z}$ since it is all ones in all indices.
- $\mathbf{1}\mathbf{1}^T$ is the infinite ones matrix
- $\mathbf{Q}$ is an infinite diagonal matrix, such that $Q_{ll} = \|k + lN\|_2^2, l \in \mathbb{Z}^m$

With these notations, we can rewrite equation 18 as:

$$\left(\lambda\mathbf{Q} + \mathbf{1}\mathbf{1}^T\right)\mathbf{u} = c\mathbf{1}. \tag{19}$$

Note that $\mathbf{Q}$ is invertible and $\mathbf{1}\mathbf{1}^T$ is of rank one. Thus, using the Sherman–Morrison matrix identity we have:

$$\mathbf{u} = c\left(\lambda\mathbf{Q} + \mathbf{1}\mathbf{1}^T\right)^{-1}\mathbf{1} = \frac{c}{\lambda}\left(\mathbf{Q}^{-1} + \mathbf{P}\right)\mathbf{1}, \tag{20}$$

where

$$P_{li} = \frac{1}{\lambda + \sum_{t\in\mathbb{Z}^m}\frac{1}{\|k+tN\|_2^2}}\frac{1}{\|k+lN\|_2^2}\frac{1}{\|k+iN\|_2^2}, \; l,i \in \mathbb{Z}^m.$$

Now, for a single unknown in $\mathbf{u}$:

$$u_l = d_{k+lN} = \frac{c}{\lambda}\frac{1}{\|k+lN\|_2^2}\left(1+\frac{\sum_{t\in\mathbb{Z}}\frac{1}{\|k+tN\|_2^2}}{\lambda+\sum_{t\in\mathbb{Z}}\frac{1}{\|k+tN\|_2^2}}\right)$$
$$\leq \frac{2c}{\lambda}\frac{1}{\|k+lN\|_2^2} = O\left(\frac{1}{\lambda\|k+lN\|_2^2}\right), \quad l\in\mathbb{Z}^m. \tag{21}$$

This is correct for any $k, l \in \mathbb{Z}$. Replacing $k+lN$ by $k \in \mathbb{Z}^m$, we have

$$d_k = O\left(\frac{1}{\lambda\|k\|_2^2}\right). \tag{22}$$

An extension of this proof to random sampling can be done by using non-orthogonal subsampled Fourier frames (see (Giryes, 2020)). □

C.2 Proof of Proposition 2

Proof. Assume $\boldsymbol{\phi}$ is represented by

$$\boldsymbol{\phi}(\mathbf{x}) = \boldsymbol{\phi}_L(\boldsymbol{\phi}_{L-1}(\cdots\boldsymbol{\phi}_2(\boldsymbol{\phi}_1(\mathbf{x};\boldsymbol{\theta}_1);\boldsymbol{\theta}_2)\cdots;\boldsymbol{\theta}_{L-1});\boldsymbol{\theta}_L), \tag{23}$$

where $\boldsymbol{\phi}_l(\cdot;\boldsymbol{\theta}_l)$ is the l-th layer with parameters $\boldsymbol{\theta}_l$, $l=1,\ldots,L$. The output of the l-th layer is denoted by $\mathbf{z}_l \in \mathbb{R}^{D_l}$, i.e. $\mathbf{z}_l \triangleq \boldsymbol{\phi}_l(\mathbf{z}_{l-1};\boldsymbol{\theta}_l)$, $l=1,\ldots,L$, and $\mathbf{z}_0 \triangleq \mathbf{x}$. Applying the chain rule to compute the network Jacobian matrix yields

$$\frac{d\boldsymbol{\phi}(\mathbf{x})}{d\mathbf{x}} = \prod_{l=1}^{L}\frac{d\mathbf{z}_l}{d\mathbf{z}_{l-1}}\,. \tag{24}$$

By using the matrix norm submultiplicativity property, we get

$$\left\|\frac{d\boldsymbol{\phi}(\mathbf{x})}{d\mathbf{x}}\right\|_2^2 = \left\|\prod_{l=1}^{L}\frac{d\mathbf{z}_l}{d\mathbf{z}_{l-1}}\right\|_2^2 \leq \prod_{l=1}^{L}\left\|\frac{d\mathbf{z}_l}{d\mathbf{z}_{l-1}}\right\|_2^2. \tag{25}$$

Now we will bound the layer Jacobian matrix spectral norm $\left\|\frac{d\mathbf{z}_l}{d\mathbf{z}_{l-1}}\right\|_2$, for various layer types and show that it can be expressed only by the weights' norm:

- *FC layer*: an FC layer is described by

$$\mathbf{z}_l = \boldsymbol{\phi}_l(\mathbf{z}_{l-1};\boldsymbol{\theta}_l) = \sigma_l(\mathbf{W}_l\mathbf{z}_{l-1}+\mathbf{b}_l), \tag{26}$$

where σ_l is the layer activation function. Hence, its Jacobian matrix is given by

$$\frac{d\mathbf{z}_l}{d\mathbf{z}_{l-1}} = \operatorname{diag}\left(\sigma_l^{'}(\mathbf{W}_l\mathbf{z}_{l-1}+\mathbf{b}_l)\right)\mathbf{W}_l. \tag{27}$$

Using the matrix norm submultiplicativity property we get

$$\left\|\frac{d\mathbf{z}_l}{d\mathbf{z}_{l-1}}\right\|_2 = \left\|\operatorname{diag}\left(\sigma_l^{'}(\mathbf{W}_l\mathbf{z}_{l-1}+\mathbf{b}_l)\right)\mathbf{W}_l\right\|_2$$
$$\leq \left\|\operatorname{diag}\left(\sigma_l^{'}(\mathbf{W}_l\mathbf{z}_{l-1}+\mathbf{b}_l)\right)\right\|_2\|\mathbf{W}_l\|_2\,. \tag{28}$$

Since the network activation functions are non-expensive, the diagonal matrix entries are not greater then 1. Hence, its spectral norm is at most 1, and we get

$$\left\|\frac{d\mathbf{z}_l}{d\mathbf{z}_{l-1}}\right\|_2 \leq \|\mathbf{W}_l\|_2\,. \tag{29}$$

Note that the commonly used activation functions such as ReLU, sigmoid and hyperbolic tangent, satisfy the non-expensive condition. Note that this proof is also relevant for a linear layer (corresponds to an identity activation, which is also non-expensive) and for convolutional layer (which can be expressed also as matrix multiplication).

- *Softmax layer:* The softmax function operation on $\mathbf{t} \in \mathbb{R}^D$ is defined by

$$\boldsymbol{\sigma}(\mathbf{t}) = \text{softmax}(\mathbf{t}) = \frac{e^{\mathbf{t}}}{\mathbf{1}^T e^{\mathbf{t}}} \ , \tag{30}$$

where the exponential is applied element-wise. Hence, its Jacobian matrix is given by

$$\frac{d\boldsymbol{\sigma}(\mathbf{t})}{d\mathbf{t}} = \text{diag}(\boldsymbol{\sigma}(\mathbf{t})) - \boldsymbol{\sigma}^T(\mathbf{t})\boldsymbol{\sigma}(\mathbf{t}). \tag{31}$$

Using the Gershgorin circle theorem, we can bound its spectral norm by

$$\begin{aligned}\left\| \frac{d\boldsymbol{\sigma}(\mathbf{t})}{d\mathbf{t}} \right\|_2 &\leq \max_{0\leq i\leq D-1} \sigma_i(\mathbf{t}) - \sigma_i^2(\mathbf{t}) + \sigma_i(\mathbf{t}) \sum_{\substack{j=0 \\ j\neq i}}^{D-1} \sigma_j(\mathbf{t}) \\ &\leq \max_{0\leq i\leq D-1} 2\sigma_i(\mathbf{t}) - \sigma_i^2(\mathbf{t}),\end{aligned} \tag{32}$$

where the last inequality is due to the fact that $\sum_{j=0}^{D-1} \sigma_j(\mathbf{t}) = 1$. The above upper bound is the maximal value of a concave parabola $p(u) = -u^2 + 2u$ in the interval $(0, 1)$, which equals 1. Hence the Jacobian matrix of a softmax layer satisfies

$$\left\| \frac{d\boldsymbol{\sigma}(\mathbf{t})}{d\mathbf{t}} \right\|_2 \leq 1. \tag{33}$$

- *Pooling layer:* A pooling layer can be written as

$$\mathbf{z}_l = \boldsymbol{\phi}_l(\mathbf{z}_{l-1}; \boldsymbol{\theta}_l) = \mathbf{P}_l(\mathbf{z}_{l-1})\mathbf{z}_{l-1}, \tag{34}$$

where $\mathbf{P}(\cdot)$ is not subject to learning. When the pooling layer operates on non-overlapping patches, the matrix representing it has orthogonal rows. In that case, the singular values are equal to the squared norm of the rows. In each row of max-pooling matrix, one entry takes the value of 1, and the rest entries equal 0. In an average pooling matrix rows, *patch_size* entries take the value of $\frac{1}{patch\_size}$, and the rest entries equal 0. Thus, in both cases, the largest singular value is smaller or equal 1, and we get

$$\left\| \frac{d\mathbf{z}_l}{d\mathbf{z}_{l-1}} \right\|_2 = \|\mathbf{P}_l(\mathbf{z}_{l-1})\|_2 \leq 1. \tag{35}$$

To summarize, we showed that a layer with parameters involved (linear and non-linear FC and convolutional layers) obeys the bound

$$\left\| \frac{d\mathbf{z}_l}{d\mathbf{z}_{l-1}} \right\|_2 \leq \|\mathbf{W}_l\|_2 \ , \tag{36}$$

and a layer with no parameters involved (softmax and pooling) obeys

$$\left\| \frac{d\mathbf{z}_l}{d\mathbf{z}_{l-1}} \right\|_2 \leq 1. \tag{37}$$

Combining these bounds with equation 25 and with the known relation between spectral and Frobenius matrix norms, we get the desired result of equation 4

□

C.3 Proof of Proposition 3

Proof. We get equation 6 using the same consideration as in the derivation of equation 39. Solving equation 6 using Karush-Kuhn-Tucker multipliers and the fact that solutions in constrained optimization tend to be on the boundary points (unless the direct solution to the ℓ_2 distance already lies within the feasible set), leads us to equation 7. □

C.4 Proof of Proposition 4

Proof. Using the derivative property and Parseval's theorem, followed by Proposition 2 and the fact that $\alpha_l = 1, l = 1, \ldots, L$, we have

$$\begin{aligned} \sum_{k\in\mathbb{Z}^m} \|k\|_2^2 \, |d_k|^2 &= \frac{1}{(2\pi)^m} \int_{x\in[0,2\pi]^m} \left\| \frac{d\phi(x)}{dx} \right\|_2^2 dx \\ &\leq \prod_{l=1}^{L} \|\mathbf{W}_l\|_2^2 = 1. \end{aligned} \tag{38}$$

By observing that $\|k\|^2 |d_k|^2 \leq \sum_{k\in\mathbb{Z}^m} \|k\|_2^2 |d_k|^2$ and plugging it on the left-hand-side of equation 38, we get that $\|k\|^2 |d_k|^2 \leq 1$. Dividing by $\|k\|^2$ leads to equation 10. □

D ASYMPTOTIC EXTENSION OF PROPOSITION 1

Proposition 5 *Let $\phi(x) = \sum_{k\in\mathbb{Z}^m} d_k e^{jk^T x}$ and $f(x) = \sum_{k\in\mathbb{Z}^m} c_k e^{jk^T x}$ be the Fourier series of the trained neural network and the target mapping function, respectively. If the training set size satisfies $N \to \infty$, then the global optimum of equation 2 is equivalent to the one of*

$$\min_{\{d_k\}_{k\in\mathbb{Z}^m}} \sum_{k\in\mathbb{Z}^m} |d_k - c_k|^2 + \lambda \sum_{k\in\mathbb{Z}^m} \|k\|_2^2 \, |d_k|^2 , \tag{39}$$

and the optimal solution reads as

$$d_k = \frac{c_k}{1 + \lambda \|k\|_2^2}, \; k \in \mathbb{Z}^m. \tag{40}$$

Proof. With a uniform sampling in the interval $[0, 2\pi]$, when $N \to \infty$, we have

$$\begin{aligned} &\frac{1}{N} \sum_{n=1}^{N} (\phi(x_n) - f(x_n))^2 \\ &\qquad \xrightarrow[N\to\infty]{} \frac{1}{(2\pi)^m} \int_{x\in[0,2\pi]^m} (\phi(x) - f(x))^2 dx \\ &\qquad = \sum_{k\in\mathbb{Z}^m} |d_k - c_k|^2, \end{aligned} \tag{41}$$

where the last equality stems from Parseval's theorem for $\phi - f$. By using the derivative property and applying Pareseval's theorem, we have

$$\frac{1}{(2\pi)^m} \int_{x\in[0,2\pi]^m} \left\| \frac{d\phi(x)}{dx} \right\|_2^2 dx = \sum_{k\in\mathbb{Z}} \|k\|_2^2 \, |d_k|^2 . \tag{42}$$

Then, equation 40 follows simply by minimizing equation 39. We can extend the proof for random sampling of the input domain by replacing the Riemann integral by a Lebesgue integral related to the sampling distribution. □

E EXPERIMENTS TECHNICAL DETAILS

All experiments were averaged over 5 trials, implemented using Tensorflow 1.15 and performed on Nvidia GeForce GTX Titan X GPU. Input pixels of the synthetic datasets were scaled to range $[0, 1]$. For Clothing1M dataset, per-pixel mean subtraction was performed.

SN. We adapt the implementation proposed in (Yoshida and Miyato, 2017; Miyato et al., 2018).

Jacobian. Instead of calculating the squared Frobenius norm of the network logits Jacobian matrix, we use an approximation of it, as proposed in (Varga et al., 2017; Hoffman et al., 2019).

F SYNTHETIC NOISE EXPERIMENTS DETAILS

F.1 Datasets, Networks and Training

F.1.1 CIFAR-10, CIFAR-100

Datasets. CIFAR-10 and CIFAR-100 datasets consist of 32x32 color images, uniformly distributed to 10 and 100 classes, respectively. The data is divided into a training set with 50,000 examples and a test set with 10,000 examples. We retained 10% from each training set for validation, and corrupted the remaining training examples, according to the uniform and flip schemes proposed in (Patrini et al., 2017). The flip noise for CIFAR-10 is described by: truck $\rightarrow$ automobile, bird $\rightarrow$ airplane, deer $\rightarrow$ horse, cat $\leftrightarrow$ dog. In CIFAR-100 the 100 classes are grouped into 20 super-classes of size 5, e.g., flowers contains orchids, poppies, roses, sunflowers, and tulips. Within each super-class, the noise flips each class into the next, circularly.

Network. For both CIFAR-10 and CIFAR-100 we used the all convolutional network (Springenberg et al., 2014), but replaced each stride 2 in the convolutional layers with max pooling with stride 2. The network consists of 9 convolutional layers: 3 of size 3x3x96, 5 of size 3x3x192 and last one of size 1x1x10, followed by global averaging and a softmax output. Max pooling is used after layers 3 and 6, and each convolution layer is followed by BN (Ioffe and Szegedy, 2015) and ReLU activation. For the baseline of the network, we used cross-entropy loss with ℓ_2 regularization, and applied early stopping (according to the validation set accuracy).

Training. The training on CIFAR datasets was performed using ADAM optimizer (Kingma and Ba, 2014) with default parameters, an initial learning rate of 0.001, a learning rate decay by a factor of 10 every 10 epochs, and a batch size of 32.

F.1.2 MNIST

Dataset. MNIST is a dataset of handwritten digits, represented by 28x28 grayscale images, which are split to a training set of size 60,000, and a test set of size 10,000. We retained 10% from the training set for validation, and corrupted the remaining training examples, according to the flip scheme. In order to further imitate realistic scenario, we used a bidirectional flip of similar classes: $1 \leftrightarrow 7$, $2 \leftrightarrow 3$, $4 \leftrightarrow 9$, $5 \leftrightarrow 6$.

Network. We used LeNet-5 (LeCun et al., 1998), where each layer (except the last layer) is batch normalized before the ReLU activation. The network consists of 2 convolutional layers of sizes 5x5x6 and 5x5x16, each followed by max pooling with stride 2; flattering layer, which vectorizes each 3-D tensor into a vector; 3 FC layers of sizes 120, 84 and 10, and a softmax output. For the baseline of the network, we used cross-entropy loss with ℓ_2 regularization, and applied early stopping (according to the validation set accuracy).

Training. The training was performed using SGD optimizer with momentum 0.9, an initial learning rate of 0.01, a learning rate decay by a factor of 10 every 15 epochs, and a batch size of 32.

F.2 Optimal Hyperparameters

The hyperparameters were tuned through the validation set. We started with searching the optimal ℓ_2 coefficient for the baseline network of each experiment, and fixed it. The search space was $\{10^{-6}, 5\cdot10^{-6}, 10^{-5}, 5\cdot10^{-5}, 10^{-4}, 5\cdot10^{-4}, 10^{-3}\}$. Then, we looked for the best Jacobian regularization configuration. First, we figured that it is better to add it after 10 epochs, rather than from the beginning. This observation stands in line with the approach of (Jakubovitz et al., 2019). Then, for each experiment, we searched the optimal coefficient out of $\{10^{-5}, 5\cdot10^{-5}, 10^{-4}, 5\cdot10^{-4}, 10^{-3}\}$, and fixed it. In the same manner, we set the entropy coefficient to be one of $\{0.5, 1, 1.5, 2\}$ and the mixup α to be in $[0.2, 0.8]$. Number of epochs ranged from 20 to 50, depending on the dataset and the noise rate. Optimal hyperparameters of all experiments are specified in Table 10, Table 11, Table 12, Table 13, and Table 14 for CIFAR-10 uniform noise, CIFAR-10 flip noise, CIFAR-100 uniform noise, CIFAR-100 flip noise, and MNIST flip noise, respectively.

Table 10: Optimal hyperparameters for CIFAR-10 with various uniform noise rates.

Regularization \ Noise Rate	0	0.1	0.3	0.5	0.7
ℓ_2	10^{-4}	10^{-4}	10^{-4}	10^{-5}	10^{-5}
Jacob	10^{-5}	10^{-4}	10^{-4}	$5 \cdot 10^{-4}$	10^{-3}
Entropy	1	2	2	1	0.5
Entropy + SN	0.5	0.5	1	1	0.5
Mixup	0.2	0.3	0.5	0.4	0.4
Epochs	20	20	20	20	30

Table 11: Optimal hyperparameters for CIFAR-10 with various flip noise rates.

Regularization \ Noise Rate	0.1	0.2	0.3	0.4	0.5
ℓ_2	10^{-4}	10^{-4}	10^{-4}	10^{-4}	10^{-4}
Jacob	10^{-5}	10^{-5}	10^{-5}	10^{-4}	10^{-4}
Epochs	20	20	20	20	20

Table 12: Optimal hyperparameters for CIFAR-100 with various uniform noise rates.

Regularization \ Noise Rate	0	0.1	0.3	0.5
ℓ_2	10^{-4}	10^{-4}	10^{-4}	$5 \cdot 10^{-5}$
Epochs	50	50	50	50

Table 13: Optimal hyperparameters for CIFAR-100 with various flip noise rates.

Regularization \ Noise Rate	0.1	0.3	0.5
ℓ_2	10^{-4}	10^{-4}	10^{-4}
Epochs	35	35	35

Table 14: Optimal hyperparameters for MNIST with various flip noise rates.

Regularization \ Noise Rate	0	0.1	0.2	0.3	0.4	0.5
ℓ_2	10^{-4}	10^{-4}	10^{-4}	10^{-4}	10^{-4}	10^{-3}
Epochs	30	30	30	30	30	30

Table 15: CIFAR-100 test accuracy for different rates of uniform noise, when trained with and without SN.

Regularization \ Noise Rate	0	0.1	0.3	0.5
ℓ_2	67.96±0.28	65.15±0.35	59.33±0.26	51.53±0.30
ℓ_2 **+ SN**	**68.75**±0.32	**66.59**±0.23	**61.23**±0.15	**52.56**±0.69

Table 16: CIFAR-100 test accuracy for different rates of flip noise, when trained with and without SN.

Regularization \ Noise Rate	0	0.1	0.3	0.5
ℓ_2	67.96±0.28	65.75±0.33	60.48±0.10	32.77±0.36
ℓ_2 **+ SN**	**68.75**±0.32	**67.03**±0.31	**63.50**±0.43	**33.13**±0.62

Table 17: MNIST test accuracy for different rates of flip noise, when trained with and without SN.

Regularization \ Noise Rate	0	0.1	0.2	0.3	0.4	0.5
ℓ_2	99.21±0.09	98.87±0.02	98.49±0.13	97.86±0.12	91.64±1.02	65.70±1.76
ℓ_2 + **SN**	**99.32**±0.08	**99.00**±0.04	**98.79**±0.07	**98.23**±0.10	**95.66**±1.09	**66.03**±1.81

Table 18: *SN combined with minimum entropy.* CIFAR-10 test accuracy when trained with different rates of uniform noise, using SN and entropy regularization.

Regularization \ Noise Rate	0	0.1	0.3	0.5	0.7
ℓ_2	90.03±0.19	87.42±0.10	83.57±0.52	79.28±0.37	69.88±0.81
ℓ_2 + **SN**	**90.82**±0.21	89.32±0.22	85.35±0.25	80.22±0.16	69.79±0.45
ℓ_2 + **Entropy**	90.07±0.16	88.33±0.18	85.45±0.19	81.27±0.56	70.97±0.50
ℓ_2 + **Entropy + SN**	90.77±0.23	**89.38**±0.14	**86.82**±0.34	**83.24**±0.12	**72.64**±0.17

G EXTENDED EXPERIMENTS

G.1 CIFAR-100

Here, we illustrate the SN effect in a more challenging task, in which there are fewer images per class. Tables 15 and 16 shows the test accuracy of the CIFAR-100 dataset when corrupted by uniform noise and flip noise, respectively. In all cases, the addition of SN increases accuracy.

G.2 MNIST

Here, we demonstrate the SN power in another network architecture, which has FC layers. The MNIST dataset is considered relatively simple. Indeed, the baseline network, and even unregularized network, show very good results for uniform noise. Flip noise introduces a small performance degradation, which, as can be seen in Table 17, is mitigated when the network weights are spectrally normalized.

H SN COMBINED WITH MINIMUM ENTROPY REGULARIZATION

Minimum entropy regularization incorporates the entropy of the network output probabilities into the loss function, and encourages the model to have high confidence in its prediction. Table 18 shows test accuracy of CIFAR-10 with different rates of uniform noise, when regularized by SN, minimum entropy, and their combination. Indeed, entropy regularization improves the accuracy, and the addition of SN increases it even more.