Learning Deeper Non-Monotonic Networks by Softly Transferring Solution Space

Zheng-Fan Wu[1,2], **Hui Xue**[1,2*] and **Weimin Bai**[1,2]

[1]School of Computer Science and Engineering, Southeast University, Nanjing, 210096, China
[2]MOE Key Laboratory of Computer Network and Information Integration (Southeast University), China
{zfwu, hxue, weiminbai}@seu.edu.cn

Abstract

Different from popular neural networks using quasiconvex activations, non-monotonic networks activated by periodic nonlinearities have emerged as a more competitive paradigm, offering revolutionary benefits: 1) compactly characterizing high-frequency patterns; 2) precisely representing high-order derivatives. Nevertheless, they are also well-known for being hard to train, due to easily over-fitting dissonant noise and only allowing for tiny architectures (shallower than 5 layers). The fundamental bottleneck is that the periodicity leads to many poor and dense local minima in solution space. The direction and norm of gradient oscillate continually during error backpropagation. Thus non-monotonic networks are prematurely stuck in these local minima, and leave out effective error feedback. To alleviate the optimization dilemma, in this paper, we propose a non-trivial soft transfer approach. It smooths their solution space close to that of monotonic ones in the beginning, and then improve their representational properties by transferring the solutions from the neural space of monotonic neurons to the Fourier space of non-monotonic neurons as the training continues. The soft transfer consists of two core components: 1) a rectified concrete gate is constructed to characterize the state of each neuron; 2) a variational Bayesian learning framework is proposed to dynamically balance the empirical risk and the intensity of transfer. We provide comprehensive empirical evidence showing that the soft transfer not only reduces the risk of non-monotonic networks on over-fitting noise, but also helps them scale to much deeper architectures (more than 100 layers) achieving the new state-of-the-art performance.

1 Introduction

Deep neural networks have led to a series of breakthroughs. Their representational properties depend heavily on the activation functions. Most activation functions typically used nowadays, e.g., Sigmoid and ReLU, are quasiconvex, mimicking the activation/inhibition of the Heaviside function.

Monotonic neuron only responding to a particular pattern makes sense from an intuitive point of view: 1) it is attracted to noticeable/generalizable/low-frequency features; 2) this monotonicity prevents complex coupling and co-adaptations between feature detectors; 3) more importantly, monotonic nonlinearity substantially smoothes the variations of gradient. In general, the solution spaces of monotonic networks are much plainer than that of non-monotonic counterparts. This plays a major role in the success of training deep networks with hundreds of millions of parameters.

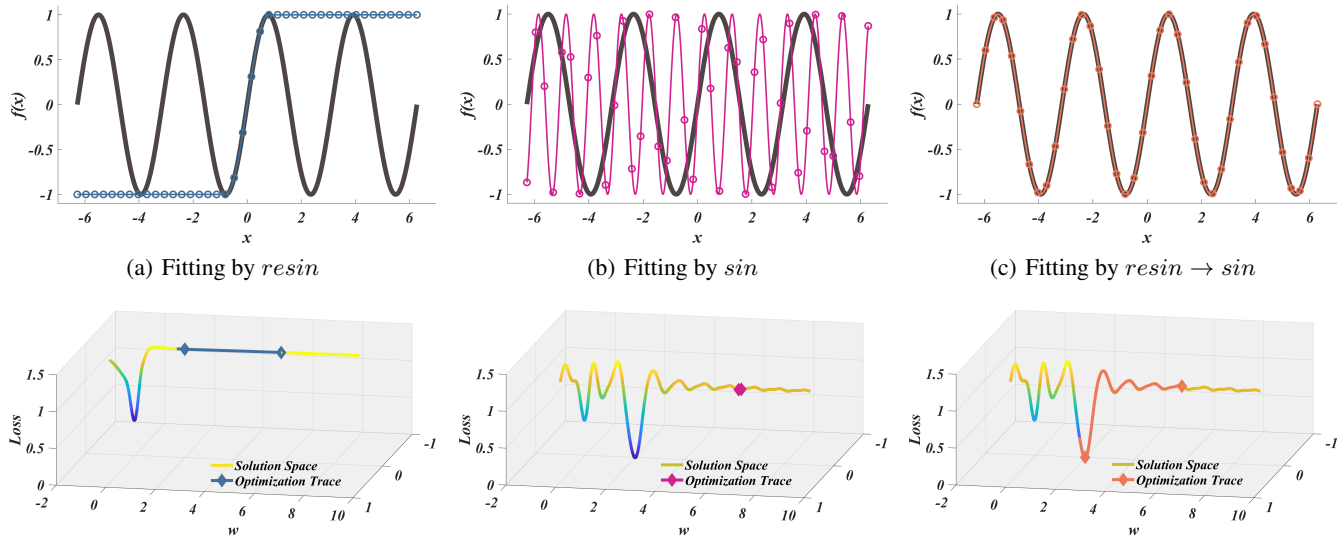
In contrary to popular monotonic ones, non-monotonic neurons are regarded as difficult to train, owing to 1) easily over-fitting and 2) only compatible with tiny networks (shallower than 5 layers). But in the last two years, we have to seriously reexamine the importance of non-monotonic activations. Many researches showed that non-monotonic activations can achieve irreplaceable effects including 1) compactly characterizing complex high-frequency patterns; 2) precisely representing implicit high-order derivatives.

Specifically, Tancik *et al.* [2020] formally demonstrated that sinusoidal Fourier mappings can dramatically perform better by allowing them to learn much higher frequencies across low-dimensional space. Sitzmann *et al.*; Bond-Taylor and Willcocks; Sitzmann *et al.* [2020b; 2020; 2020a] comprehensively proved the great power of sinusoidal units in modelling complex signals with fine detail and implicit derivatives. Xue *et al.*; Xue and Wu [2019; 2020] used the sinusoidal function to reveal dynamic characteristics and potential correlations. Mildenhall *et al.*; Zhong *et al.* [2020; 2019] improved the state-of-the-art performance on the tasks of novel view synthesis by using sinusoids in Fourier space.

However, these non-monotonic networks built on sinusoidal nonlinearities are paired with manually initialization and tiny architectures (shallower than 5 layers). Although the important properties of non-monotonic networks are recognized, their intractable optimization dilemma left over by history has not yet been solved.

On the one hand, they are more inclined to over-fit noise hidden in signals. Their periodic neurons can be activated across the whole feature space. As the correlation with the input increases, the activation will oscillate between stronger and weaker, and thus is more attracted to dissonant noise. On

*Contact Author



(a) Fitting by $resin$ (b) Fitting by sin (c) Fitting by $resin \rightarrow sin$

(d) Optimizing $resin$ (e) Optimizing sin (f) Optimizing $resin \rightarrow sin$

Figure 1: The results on fitting $\sin(2x), x \in [-2\pi, 2\pi]$, and the corresponding optimization traces in solution space.

the other hand, they can hardly work well in more complex architectures. Non-monotonic neuron responds to multiple patterns, and easily couples and co-adapts with each other. The direction and norm of gradient oscillate continually in a local range during error backpropagation. Therefore, it is a great challenge to optimize deep non-monotonic networks.

Actually, even a network with only one single sinusoidal neuron has infinite VC dimension. The periodicity gives rise to numerous poor and dense local minima in solution space. Non-monotonic networks may be prematurely stuck in poor local minima, and leave out more effective error feedback. Hence, we pay particular attention to smoothing their solution space by taking monotonic variants as the helper, so as to help non-monotonic networks learn low/high-frequency concepts better and scale them to substantially deeper and wider senior architectures.

As a first step toward understanding this optimization dilemma intuitively, we conduct a synthetic experiment on fitting the scalar $\sin(2x), x \in [-2\pi, 2\pi]$ with three networks defined in Eq. (1): 1) the monotonically-rectified sinusoidal network $\bar{\sigma} := resin$; 2) the non-monotonic $\tilde{\sigma} := sin$; 3) the non-monotonic $\sigma_z := resin \rightarrow sin$ softly transferring the solution from $resin$ to sin. They have only one neuron paired with a scalar weight w initialized at $w = 6$. The fitting results and optimization processes are shown in Figure 1.

Firstly, the solution space of $resin$ is much smoother than that of sin. But $resin$ can only capture the local features on $x \in [-\frac{\pi}{4}, \frac{\pi}{4}]$. Even if it is optimized to the global optimum $w = 0$, $resin$ just outputs 0 constantly owing to the structural limitation. Secondly, in contrast, the solution space of the non-monotonic sin has more poor and dense local minima. sin is prematurely stuck in the terrible solution and fits the wrong frequency. Thirdly, $resin \rightarrow sin$ finds the optimal solution $w = 2$ and perfectly reconstructs the original function. Consequently, by accurately transferring the solution from monotonic network to the associated non-monotonic network, we can alleviate the optimization difficulty on the premise of preserving the structural superiority.

In this paper, we propose a non-trivial soft transfer approach, bridging the gap between the neural solution space of monotonic networks and the Fourier solution space of non-monotonic networks. The novel transfer alleviates the optimization dilemma by smoothing their solution space close to that of monotonic counterparts in the early stage of training, and then improves their representational properties by transferring the solutions from the monotonic nonlinearities to the non-monotonic ones as the training continues. Specifically, the soft transfer is implemented by two core components:

- To automatically characterize the monotonicity/non-monotonicity of each neuron, a parametric gate with the rectified concrete distribution is constructed.
- To dynamically balance the empirical risk and the intensity of transfer, a more efficient learning framework based on variational Bayesian inference is proposed.

By taking the soft transfer approach, non-monotonic networks are as easy to optimize as their monotonic counterparts. Moreover, we provide comprehensive empirical evidence showing that:

- Soft transfer reduces the risk of over-fitting noise. Low-frequency concepts are chiefly learned by monotonic parts, and then high-frequency details and high-order derivatives are represented by non-monotonic parts.
- Soft transfer helps non-monotonic networks scale to deeper and wider architectures. The solution space changes smoothly from simple to complex in training, without sacrificing any representational property.

We perform various experiments to demonstrate the effectiveness of the soft transfer. Based on the proposed approach,

non-monotonic networks are successfully extended to the senior residual learning architectures deeper than 100 layers, and achieve the new state-of-the-art performance.

2 Related Work

On the one hand, Tancik *et al.* [2020] comprehensively showed that standard neural networks are poorly suited for learning high-frequency content, a phenomenon referred to spectral bias caused by a rapid frequency falloff. Sitzmann *et al.*; Bond-Taylor and Willcocks; Sitzmann *et al.* [2020b; 2020; 2020a] proved that monotonic activations are incapable of modeling information contained in higher-order derivatives of natural signals.

On the other hand, more important properties of non-monotonic networks have been recognized. Prior works in natural language processing and time series analysis have used sinusoidal functions as positional encoding [Vaswani *et al.*, 2017; Xu *et al.*, 2019; Kazemi *et al.*, 2019]. Xue *et al.*; Xue and Wu [2019; 2020; 2017] pointed out that periodic nonlinearities have the potential to reveal input-dependent and long-range characteristics. Coordinate-based neural networks paired with sinusoidal activations achieved the new state-of-the-art performance on novel view synthesis [Mildenhall *et al.*, 2020; Zhong *et al.*, 2019].

But as yet, an available approach solving their optimization dilemma left over by history is still missing.

3 Softly Transferring Solution Space

This section contains three main parts: 1) a naïve but intractable recipe; 2) a variational Bayesian learning framework; 3) a rectified concrete gate. In the end, some distinctive properties of this soft transfer scheme are analyzed.

Above all, we make an innocuous stipulation that non-monotonic networks are composed of periodic neurons activated by sinusoidal functions. But it should be emphasized that the soft transfer approach is naturally compatible with other variants without modification.

3.1 Non-Monotonicity Gated Unit

A non-monotonicity gated unit is defined to characterize the monotonicity/non-monotonicity of each neuron. Consider the general activation that can be either monotonic or not.

$$\sigma_z(\cdot) = z \odot \bar{\sigma}(\cdot) + (1 - z) \odot \tilde{\sigma}(\cdot), \quad z \in \{0, 1\}. \tag{1}$$

$\tilde{\sigma}(\cdot)$ is a non-monotonic activation and $\bar{\sigma}(\cdot)$ is its monotonic variant. Without loss of generality, we mainly consider the widely-used sinusoidal function $\tilde{\sigma}(x) = \sin(x)$ and its monotonically-rectified variant $\bar{\sigma}(x) = resin(x)$, with respect to an input x, where $resin(x)$ is defined by

$$resin(x) = \begin{cases} -1, & x \in (-\infty, -\frac{\pi}{2}] \\ \sin(x), & x \in (-\frac{\pi}{2}, \frac{\pi}{2}) \\ 1, & x \in [\frac{\pi}{2}, \infty) \end{cases}. \tag{2}$$

Whether $\sigma_z(\cdot)$ is monotonic or periodic is controlled by the discrete value of the binary gate z.

By initializing all gates z in a network to 1 and penalizing those gates z for being different than exact 0 as the training continues, we can mimic the behavior of transferring the solution space from a monotonic neural network to a non-monotonic Fourier network. The transfer has three meanings: 1) the nonlinearity is switched from monotonic to periodic; 2) the representational property is changed from low-frequency to high-frequency; 3) the solution space is transformed from smooth/neural to bumpy/Fourier.

Controlling the proportion of different kinds of neurons properly, we can ease the complexity of the solution space in the beginning and improve the representational properties as the training continues. But the practical optimization under this transfer is computationally intractable due to the non-differentiability and the combinatorial nature of $2^{|\sigma_z|}$ possible states, where $|\sigma_z|$ is the total number of activations.

3.2 Differentiable Soft Transfer Framework

For this reason, we propose a more efficient differentiable learning framework for softly transferring solution space, utilizing variational Bayesian inference as its theoretical basis.

Given some observed data $\mathcal{D}$, a group of random variables $\boldsymbol{z}$ gating monotonicity/non-monotonicity, and a collection of activations $\boldsymbol{\sigma_z}$ regarded as random variables reparameterized by $\boldsymbol{z}$. According to Bayesian inference, a general learning problem can be formalized by a log-probability $\log \mathbb{P}(\mathcal{D})$.

$$\begin{aligned} \log \mathbb{P}(\mathcal{D}) &= \log \mathbb{P}(\mathcal{D}, \boldsymbol{\sigma_z}, \boldsymbol{z}) - \log \mathbb{P}(\boldsymbol{\sigma_z}, \boldsymbol{z}|\mathcal{D}) \\ &= \underbrace{\int \log \frac{\mathbb{P}(\mathcal{D}, \boldsymbol{\sigma_z}, \boldsymbol{z})}{q(\boldsymbol{\sigma_z}, \boldsymbol{z})} q(\boldsymbol{\sigma_z}, \boldsymbol{z}) d\boldsymbol{\sigma_z} d\boldsymbol{z}}_{\mathcal{L}(\mathcal{D}, \boldsymbol{\sigma_z}, \boldsymbol{z})} + \\ &\quad \underbrace{\int \log \frac{q(\boldsymbol{\sigma_z}, \boldsymbol{z})}{\mathbb{P}(\boldsymbol{\sigma_z}, \boldsymbol{z}|\mathcal{D})} q(\boldsymbol{\sigma_z}, \boldsymbol{z}) d\boldsymbol{\sigma_z} d\boldsymbol{z}}_{KL(q(\boldsymbol{\sigma_z}, \boldsymbol{z}) \| \mathbb{P}(\boldsymbol{\sigma_z}, \boldsymbol{z}|\mathcal{D}))}. \end{aligned} \tag{3}$$

q is the introduced approximate posterior over $\boldsymbol{\sigma_z}$ and $\boldsymbol{z}$. Moreover, because $\log \mathbb{P}(\mathcal{D})$ is a constant if $\mathcal{D}$ is given, maximizing the Evidence Lower BOund (ELBO) $\mathcal{L}(\mathcal{D}, \boldsymbol{\sigma_z}, \boldsymbol{z})$ is equivalent to minimizing the Kullback–Leibler divergence (KL-divergence) $KL(q(\boldsymbol{\sigma_z}, \boldsymbol{z}) \| \mathbb{P}(\boldsymbol{\sigma_z}, \boldsymbol{z}|\mathcal{D}))$. Generally, we consider minimizing the negative ELBO $-\mathcal{L}(\mathcal{D}, \boldsymbol{\sigma_z}, \boldsymbol{z})$.

Furthermore, suppose p is a spike and slab prior over $\boldsymbol{\sigma_z}$ and $\boldsymbol{z}$, which is the widely-used golden standard in selecting variables as far as Bayesian inference is concerned. It is defined as a mixture of a delta spike at zero and a continuous distribution over the real line.

$$\begin{aligned} &p(\boldsymbol{z}) = \text{Bernoulli}(\boldsymbol{\lambda}), \\ &p(\boldsymbol{\sigma_z}|\boldsymbol{z} = \mathbf{0}) = \delta(\boldsymbol{\sigma_z}), \\ &p(\boldsymbol{\sigma_z}|\boldsymbol{z} \neq \mathbf{0}) = \mathcal{N}(\boldsymbol{\sigma_z}|\mathbf{0}, \mathbf{1}). \end{aligned} \tag{4}$$

Since the true posterior distribution under this prior is intractable, we let $q(\boldsymbol{\sigma_z}, \boldsymbol{z})$ be a spike and slab approximate posterior over $\boldsymbol{\sigma_z}$ and $\boldsymbol{z}$. The negative ELBO $-\mathcal{L}(\mathcal{D}, \boldsymbol{\sigma_z}, \boldsymbol{z})$ under the spike and slab prior and approximate posterior over $\boldsymbol{\sigma_z}$ and $\boldsymbol{z}$ can be rewrited as

$$\begin{aligned} &-\mathcal{L}(\mathcal{D}, \boldsymbol{\sigma_z}, \boldsymbol{z}) \\ &= -\mathbb{E}_{q(\boldsymbol{\sigma_z}, \boldsymbol{z})} \big[\log \mathbb{P}(\mathcal{D}|\boldsymbol{\sigma_z}, \boldsymbol{z}) \big] + KL(q(\boldsymbol{\sigma_z}, \boldsymbol{z}) \| p(\boldsymbol{\sigma_z}, \boldsymbol{z})). \end{aligned} \tag{5}$$

We assume that the non-monotonicity gated units are independent of each other. That is, p and q factorize over the

dimensionality of $\boldsymbol{\sigma_z}$ and $\boldsymbol{z}$ in an element-wise way. Furthermore, according to the chain rule of KL-divergence, we have

$$\begin{aligned}
&-\mathcal{L}(\mathcal{D},\boldsymbol{\sigma_z},\boldsymbol{z})\\
=&-\mathbb{E}_{q(\boldsymbol{z})q(\boldsymbol{\sigma_z}|\boldsymbol{z})}\big[\log\mathbb{P}(\mathcal{D}|\boldsymbol{\sigma_z})\big]+\sum_{i=1}^{|\boldsymbol{\sigma_z}|}KL(q(z_i)\|p(z_i))\\
&+\sum_{i=1}^{|\boldsymbol{\sigma_z}|}q(z_i=0)KL(q(\sigma_{z,i}|z_i=0)\|p(\sigma_{z,i}|z_i=0))\\
&+\sum_{i=1}^{|\boldsymbol{\sigma_z}|}q(z_i\neq 0)KL(q(\sigma_{z,i}|z_i\neq 0)\|p(\sigma_{z,i}|z_i\neq 0)).
\end{aligned}\tag{6}$$

Since

$$\begin{aligned}
&KL(q(z_i)\|p(z_i))\geq 0,\\
&KL(q(\sigma_{z,i}|z_i=0)\|p(\sigma_{z,i}|z_i=0))=0,\\
&KL(q(\sigma_{z,i}|z_i\neq 0)\|p(\sigma_{z,i}|z_i\neq 0))=\gamma,
\end{aligned}\tag{7}$$

where γ is a weighting factor for explicitly penalizing monotonic neurons for being different than non-monotonic ones, $-\mathcal{L}(\mathcal{D},\boldsymbol{\sigma_z},\boldsymbol{z})$ can be further represented as

$$\begin{aligned}
&-\mathcal{L}(\mathcal{D},\boldsymbol{\sigma_z},\boldsymbol{z})\\
\geq&-\mathbb{E}_{q(\boldsymbol{z})q(\boldsymbol{\sigma_z}|\boldsymbol{z})}\big[\log\mathbb{P}(\mathcal{D}|\boldsymbol{\sigma_z})\big]+\gamma\sum_{i=1}^{|\boldsymbol{\sigma_z}|}q(z_i\neq 0).
\end{aligned}\tag{8}$$

As long as we apply a differentiable approximate posterior $q(\boldsymbol{z}|\phi)$ allowing for the reparameterization trick $\boldsymbol{z}=f(\boldsymbol{\phi},\boldsymbol{\epsilon})$ over the parameters $\boldsymbol{\phi}$ and a parameter free noise distribution $\tau(\boldsymbol{\epsilon})$, we can reformulate the optimization objective $-\mathcal{L}(\mathcal{D},\boldsymbol{\sigma_z},\boldsymbol{z})$ and solve it by Monte Carlo approximation.

$$\begin{aligned}
&-\mathcal{L}(\mathcal{D},\boldsymbol{\sigma_z},\boldsymbol{z})\\
\geq&-\mathbb{E}_{\tau(\boldsymbol{\epsilon})}\big[\log\mathbb{P}(\mathcal{D}|\boldsymbol{\sigma}_{\boldsymbol{f}(\boldsymbol{\phi},\boldsymbol{\epsilon})})\big]+\gamma\sum_{i=1}^{|\boldsymbol{\sigma_z}|}q(z_i\neq 0|\phi_i),\\
\approx&-\sum_{k=1}^{K}\log\mathbb{P}(\mathcal{D}|\boldsymbol{\sigma}_{\boldsymbol{f}(\boldsymbol{\phi},\boldsymbol{\epsilon}^{(k)})})+\gamma\sum_{i=1}^{|\boldsymbol{\sigma_z}|}q(z_i\neq 0|\phi_i).
\end{aligned}\tag{9}$$

Crucially, the learning objective is now differentiable with respect to the parameters $\boldsymbol{\phi}$, thus enabling for efficient stochastic gradient based optimization. The parameters of the distribution over the gates can then be jointly optimized with the original network parameters.

3.3 Parametric Gate with Concrete Distribution

Based on the differentiable learning framework, we further refine the parametric gate by utilizing a continuously differentiable distribution allowing for the reparameterization trick. Assume that we have a binary concrete random variable v distributed in the $(0,1)$ interval with probability density function $q_v(v|\phi)$ and cumulative distribution function $Q_v(v|\phi)$. The parameters of the distribution are $\phi=(\alpha,\beta)$, where $\log\alpha$ is the location and $0<\beta<1$ is the temperature, controlling the degree of approximate Bernoulli distribution. We have

$$v=\text{Sigmoid}\big((\log\alpha+\log\epsilon-\log(1-\epsilon))\beta^{-1}\big),\tag{10}$$

where $\epsilon\sim\mathcal{U}(0,1)$. We can calculate $q_v(v|\phi)$ and $Q_v(v|\phi)$ analytically.

$$\begin{aligned}
q_v(v|\phi)=&\frac{\alpha\beta v^{-\beta-1}(1-v)^{-\beta-1}}{(\alpha v^{-\beta}+(1-v)^{-\beta})^2},\\
Q_v(v|\phi)=&\text{Sigmoid}\big(\log(\frac{v}{1-v})\beta-\log\alpha\big).
\end{aligned}\tag{11}$$

Here, we stretch the binary concrete distribution to the (ξ,ζ) interval, with $\xi\leq 0$ and $\zeta\geq 1$, and rectify it in $[0,1]$ by applying a min-max transformation.

$$\begin{aligned}
\breve{v}=&v(\zeta-\xi)+\xi,\\
z=&\min(1,\max(0,\breve{v})).
\end{aligned}\tag{12}$$

This would then induce a distribution where the probability mass of $q_{\breve{v}}(\breve{v}|\phi)$ on the negative values $Q_{\breve{v}}(0|\phi)$, is folded to a delta peak at zero, the probability mass on values larger than one, $1-Q_{\breve{v}}(1|\phi)$ is folded to a delta peak at one, and the original distribution $q_{\breve{v}}(\breve{v}|\phi)$ is truncated to the $(0,1)$ interval. More details are referred to the concrete distribution [Maddison *et al.*, 2016; Louizos *et al.*, 2018].

Furthermore, considering $q(z\neq 0|\phi)=1-Q_{\breve{v}}(\breve{v}\leq 0|\phi)$, we define the general optimization objective by minimizing the total risk $\mathcal{R}(\mathcal{D})$.

$$\mathcal{R}(\mathcal{D}):=-\log\mathbb{P}(\mathcal{D}|\boldsymbol{\sigma_z})+\gamma\sum_{i=1}^{|\boldsymbol{\sigma_z}|}\big[1-Q_{\breve{v}_i}(0|\phi_i)\big],\tag{13}$$

where

$$1-Q_{\breve{v}_i}(0|\phi_i)=\text{Sigmoid}\big(\log\alpha_i-\beta\log(-\frac{\xi}{\zeta})\big).\tag{14}$$

In training, z can be sampled efficiently.

$$\begin{aligned}
\epsilon\sim&\mathcal{U}(0,1),\\
v=&\text{Sigmoid}\big((\log\alpha+\log(\frac{\epsilon}{1-\epsilon}))\beta^{-1}\big),\\
\breve{v}=&v(\zeta-\xi)+\xi,\\
z=&\min(1,\max(0,\breve{v})).
\end{aligned}\tag{15}$$

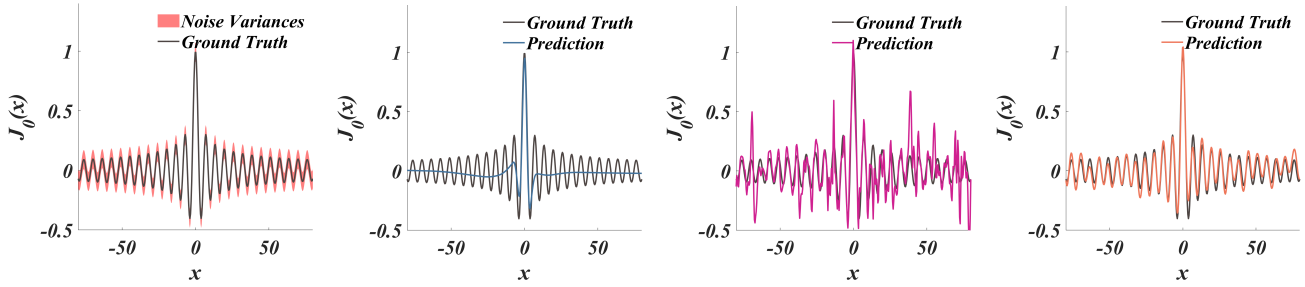
In prediction, we apply the following estimator.

$$\bar{z}=\min(1,\max(0,\text{Sigmoid}(\log\alpha)(\zeta-\xi)+\xi)).\tag{16}$$

The total risk $\mathcal{R}(\mathcal{D})$ is a special case of the negative ELBO $-\mathcal{L}(\mathcal{D},\boldsymbol{\sigma_z},\boldsymbol{z})$ by setting the sampling number $K=1$. The reason for optimizing $\mathcal{R}(\mathcal{D})$ is that we focus on optimizing large-scale networks efficiently instead of reducing the uncertainty of $\boldsymbol{z}$. As the training continues, these random variables $\boldsymbol{z}$ will be penalized close to zero.

3.4 Analysis

Following the above theoretical inference, the application of soft transfer scheme is quite flexible. The general optimization objective $\mathcal{R}(\mathcal{D})$ can be reformulated as 1) the empirical risk $-\log\mathbb{P}(\mathcal{D}|\boldsymbol{\sigma_z})$ and 2) the structural risk $\sum_{i=1}^{|\boldsymbol{\sigma_z}|}\big[1-Q_{\breve{v}_i}(0|\phi_i)\big]$. The former empirical risk is calculated by a conventional loss function characterizing how well the model fits the observed data. What we need to do is add the latter structural risk to the loss as an additional regularizer, and replace



(a) Ground Truth (b) $resin$ (c) sin (d) $resin \rightarrow sin$

Figure 2: Learning the Bessel function with the white Gaussian noise.

the activation with the non-monotonicity gated unit $\boldsymbol{\sigma_z}$ parameterized by $\boldsymbol{\phi}$. $\boldsymbol{\phi}$ can be shared by different neurons depending on the transfer granularity. Even in the most fine-grained case, the total number of $\boldsymbol{\phi}$ does not exceed the sum of the output dimensions in all neurons, which is far less than that of the original network parameters.

In addition to smoothing the solution space to ease the optimization difficulty, soft transfer actually has some other important advantages. Firstly, it is essentially a regularizer that balances the degree of learning observed data and the proportion of non-monotonicity. Secondly, it enables the proper uncertainty for the networks and then encourages the optimizer to explore more potential solutions in a space $2^{|\sigma_z|}$ larger than the original. Thirdly, it prevents complex coupling and co-adaptations between feature detectors by switching the monotonic/non-monotonic state of each neuron.

4 Experiments

4.1 Experimental Networks

- sin: Sinusoidal activation as the most important baseline.
- $resin$: Monotonically-rectified sinusoidal activation.
- $tanh$: Tanh activation that is the smooth approximation of the discontinuous $resin$.
- $relu$: Hugely popular ReLU activation.
- $Copy$: Directly copying the parameters from $resin$ trained in the first half of epoches to sin, and then continue to train sin in the second half of epoches.
- $Line$: Linearly penalizing the $\boldsymbol{z}$ from 1 to 0.
- $Bern$: $\boldsymbol{z}$ is subject to a Bernoulli distribution.
- $resin \rightarrow sin$: Soft transfer from $resin$ to sin.
- $tanh \rightarrow sin$: Soft transfer from $tanh$ to sin.
- $relu \rightarrow sin$: Soft transfer from $relu$ to sin.

4.2 Experimental Settings

The weights and biases of sin are respectively initialized by $\mathcal{N}(0, 0.1)$ and $\mathcal{U}(-\pi, \pi)$ in accordance with the explanation in [Rahimi and Recht, 2008; Xue *et al.*, 2019]. Other networks are initialized according to the Kaiming method [He *et al.*, 2016]. They are all optimized by SGD with a mini-batch size of 128, a weight decay of 10^{-4}, and a Nesterov momentum of 0.9 [Paszke *et al.*, 2019; Sutskever *et al.*, 2013; Goodfellow *et al.*, 2016]. The learning rate is initially set to 0.1, and then it is adjusted by a cosine annealing schedule with warm restarts [Loshchilov and Hutter, 2016]. For all soft transfer related networks, we set $\beta = \frac{2}{3}$, $\xi = -0.1$, and $\zeta = 1.1$ following the recommendations [Maddison *et al.*, 2016; Jang *et al.*, 2016]. α is initialized by sampling from $\mathcal{N}(8, 1)$.

4.3 Learning the Bessel Function

To evaluate that the proposed soft transfer can reduce the risk of over-fitting high-frequency noise, we conduct an experiment to learn the first kind of 0-order Bessel function $J_0(x), x \in [-80, 80]$, which contains a lot of implicit high-frequency features and high-order derivatives. We uniformly get 400 samples $\{(x_i, J_0(x_i))\}_{i=1}^{400}$, and randomly divide them into two non-overlapping training and test sets that are equal in size. The extra white Gaussian noise $\mathcal{N}(0, 0.08^2)$ is added to the training labels for simulating the interference in practical tasks. $resin$, sin, and $resin \rightarrow sin$ are used to fit $J_0(x)$. They have the same fully-connected $400 \times 400 \times 400$ architecture. The results are shown in Figure 2.

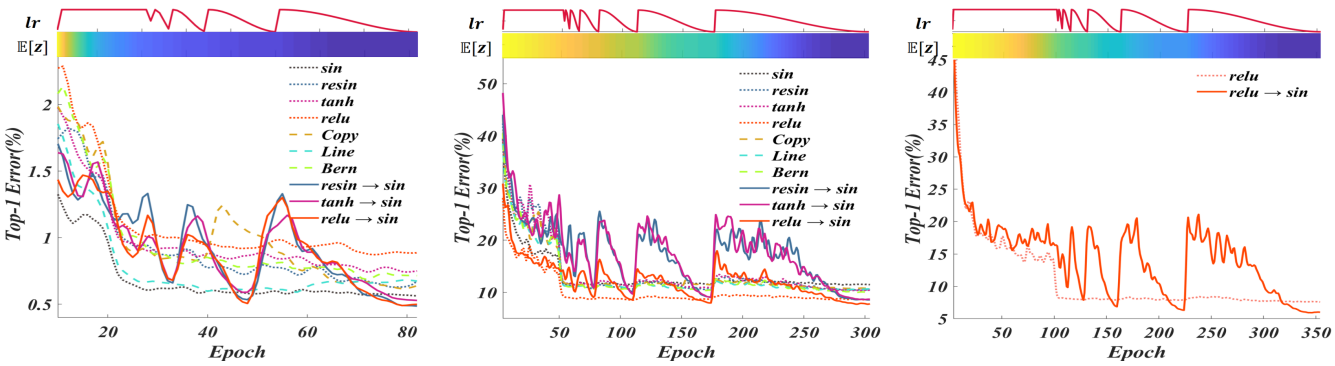
Owing to the structural limitation, The insensitive $resin$ makes a very conservative decision and outputs the average value in the high-frequency range. But the periodic sin is over-sensitive to learn the white noise added to each sample. This is a possible reason why non-monotonic networks are inclined to make over-confident but inaccurate decisions in practical tasks. In contrast, $resin \rightarrow sin$ with the soft transfer scheme achieves the best fitting result. The low-frequency concepts are chiefly guaranteed to be learned well by the initial monotonic parts, and then high-frequency details and high-order derivatives are further extracted by the transferred non-monotonic neurons. $resin \rightarrow sin$ not only accurately captures the key information such as frequency and phase, but also resists the noisy interference.

4.4 Learning Image Classification

We conduct image experiments to learn **MNIST** [LeCun *et al.*, 1998] and **CIFAR10** [Krizhevsky *et al.*, 2009] by the shallower LeNet-5 architecture [LeCun *et al.*, 1998] and the deeper ResNet-20/110 architectures, respectively. The division of datasets is consistent with their default settings. The results are presented in Table 1 and Figure 3.

Shallower LeNet-5 Architecture

On this simple task paired with the shallow LeNet-5 architecture, all networks associated with sin perform better than



(a) **MNIST** (LeNet-5) (b) **CIFAR10** (ResNet-20) (c) **CIFAR10** (ResNet-110)

Figure 3: Classification top-1 error(%) on the image datasets. Different curves represent the learning processes of different networks where soft transfer based ones are denoted as the best solid lines. The learning rate lr and the degree of transfer $\mathbb{E}[\boldsymbol{z}]$ are also shown at the top.

Top-1 Error (%)	MNIST (LeNet-5) conv	MNIST (LeNet-5) best	CIFAR10 (ResNet-20) conv	CIFAR10 (ResNet-20) best	CIFAR10 (ResNet-110) conv	CIFAR10 (ResNet-110) best
sin	0.56	0.53	11.39	11.25	-	-
$resin$	0.73	0.60	10.41	10.30	-	-
$tanh$	0.76	0.72	10.48	10.33	-	-
$relu$	0.89	0.86	8.63	8.47	7.55	7.49
$Copy$	0.66	0.61	10.94	10.66	-	-
$Line$	0.66	0.55	10.59	10.41	-	-
$Bern$	0.70	0.66	10.10	9.85	-	-
$resin \rightarrow sin$	**0.48**	0.48	8.53	8.50	-	-
$tanh \rightarrow sin$	0.52	0.49	8.67	8.54	-	-
$relu \rightarrow sin$	0.50	**0.47**	**7.94**	**7.55**	**5.99**	**5.84**

Table 1: Classification top-1 error(%) on the image datasets. **conv** means the convergent error in the last epoch and **best** means the best error in all epochs. The best results are highlighted in **bold**.

monotonic ones. The sinusoidal neuron compactly represents implicit details. In general, the non-monotonic networks built on the soft transfer ($resin \rightarrow sin$, $tanh \rightarrow sin$, and $relu \rightarrow sin$) achieve the most competitive performance 0.47%, owing to the preferable balance between the empirical risk and the proportion of non-monotonicity.

Deeper ResNet-20/110 Architectures

First, it is completely opposite to the results presented in the MNIST classification that these networks associated with sin but not built on the soft transfer approach perform worst. Once the network architecture becomes a little bit complex, $\sin$ can not make its excellent theoretical properties yield well as we expected. The three naïve transfer scheme ($Copy$, $Line$, and $Bern$) do not work at all. In contrast, the soft transfer based networks fully realize the great power of $\sin$ through cautiously transferring the solution from the monotonic nonlinearity to the sinusoidal nonlinearity. $relu \rightarrow sin$ successfully achieve the best top-1 error 7.55% that is also competitive compared with the record 7.51% of the deeper ResNet-32 in the publication [He *et al.*, 2016].

Second, interestingly, it may not be a tough requirement for softly transferring that the source neuron and the target neuron have some similar mapping structures. According to the performance of $relu \rightarrow sin$, the knowledge is successfully transferred from the neural space of the piecewise linear $relu$ to the Fourier space of the periodic $\sin$, which implies that the soft transfer is a very general and flexible framework and has more potential to be developed.

Last, based on the soft transfer approach and the monotonic ReLU helper, we successfully train the non-monotonic network $relu \rightarrow sin$ in the ResNet-110 architecture. The best top-1 error 7.49% of $relu$ is slightly higher than that of the official record 6.61%. But $relu \rightarrow sin$ still achieve the best performance 5.84%. Paired with the non-trivial soft transfer, the superiority of the non-monotonic $\sin$ in representation are well coordinated with the advantages of the monotonic $relu$ in gradient propagation. Other networks failed to train under this giant architecture due to the well-known gradient vanishing/explosion, and $\sin$ behaves as poorly as random guessing.

5 Conclusion

Despite the increasing emergence of non-monotonic networks, an available approach solving their optimization dilemma is still missing. In this paper, we propose a novel soft transfer approach consisting of two core components: 1) a rectified concrete gate is constructed to characterize the state of each neuron; 2) a variational Bayesian learning framework is proposed to dynamically balance the empirical risk and the degree of transfer. Consequently, it can help non-monotonic networks learn low/high-frequency concepts better and scale them to substantially deeper and wider senior architectures. Systematical experiments demonstrate the effectiveness of the proposed soft transfer approach.

Acknowledgments

This work was supported by National Natural Science Foundation of China (Grant No.62076062) and National Key R&D Program of China (Grant No.2017YFB1002801). Furthermore, the work was also supported by Collaborative Innovation Center of Wireless Communications Technology.

References

[Bond-Taylor and Willcocks, 2020] Sam Bond-Taylor and Chris G Willcocks. Gradient origin networks. *arXiv preprint arXiv:2007.02798*, 2020.

[Goodfellow *et al.*, 2016] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT press, 2016.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[Jang *et al.*, 2016] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

[Kazemi *et al.*, 2019] Seyed Mehran Kazemi, Rishab Goel, Sepehr Eghbali, Janahan Ramanan, Jaspreet Sahota, Sanjay Thakur, Stella Wu, Cathal Smyth, Pascal Poupart, and Marcus Brubaker. Time2vec: Learning a vector representation of time. *arXiv preprint arXiv:1907.05321*, 2019.

[Krizhevsky *et al.*, 2009] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[LeCun *et al.*, 1998] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[Loshchilov and Hutter, 2016] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

[Louizos *et al.*, 2018] Christos Louizos, Max Welling, and Diederik P Kingma. Learning sparse neural networks through l0 regularization. In *International Conference on Learning Representations*, 2018.

[Maddison *et al.*, 2016] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, 2016.

[Mildenhall *et al.*, 2020] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *arXiv preprint arXiv:2003.08934*, 2020.

[Paszke *et al.*, 2019] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*, 2019.

[Rahimi and Recht, 2008] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, pages 1177–1184, 2008.

[Ramachandran *et al.*, 2017] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.

[Sitzmann *et al.*, 2020a] Vincent Sitzmann, Eric R Chan, Richard Tucker, Noah Snavely, and Gordon Wetzstein. Metasdf: Meta-learning signed distance functions. *arXiv preprint arXiv:2006.09662*, 2020.

[Sitzmann *et al.*, 2020b] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33, 2020.

[Sutskever *et al.*, 2013] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147, 2013.

[Tancik *et al.*, 2020] Matthew Tancik, Pratul P Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *arXiv preprint arXiv:2006.10739*, 2020.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[Xu *et al.*, 2019] Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. Self-attention with functional time representation learning. In *Advances in Neural Information Processing Systems*, pages 15915–15925, 2019.

[Xue and Wu, 2020] Hui Xue and Zheng-Fan Wu. Bakernets: Bayesian random kernel mapping networks. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3073–3079. International Joint Conferences on Artificial Intelligence Organization, 7 2020.

[Xue *et al.*, 2019] Hui Xue, Zheng-Fan Wu, and Wei-Xiang Sun. Deep spectral kernel learning. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 4019–4025. International Joint Conferences on Artificial Intelligence Organization, 7 2019.

[Zhong *et al.*, 2019] Ellen D Zhong, Tristan Bepler, Joseph H Davis, and Bonnie Berger. Reconstructing continuous distributions of 3d protein structure from cryo-em images. In *International Conference on Learning Representations*, 2019.