Improving Robustness via Tilted Exponential Layer: A Communication-Theoretic Perspective

Bhagyashree Puranik[1] **Ahmad Beirami**[2] **Yao Qin**[1,2] **Upamanyu Madhow**[1]
[1]University of California Santa Barbara [2]Google Research

Abstract

State-of-the-art techniques for enhancing robustness of deep networks mostly rely on empirical risk minimization with suitable data augmentation. In this paper, we propose a complementary approach motivated by communication theory, aimed at enhancing the *signal-to-noise ratio* at the output of a neural network layer via neural competition during learning and inference. In addition to standard empirical risk minimization, neurons compete to sparsely represent layer inputs by maximization of a tilted exponential (TEXP) objective function for the layer. TEXP learning can be interpreted as maximum likelihood estimation of *matched filters* under a Gaussian model for *data noise*. Inference in a TEXP layer is accomplished by replacing batch norm by a tilted softmax, which can be interpreted as computation of posterior probabilities for the competing signaling hypotheses represented by each neuron. After providing insights via simplified models, we show, by experimentation on standard image datasets, that TEXP learning and inference enhances robustness against noise and other common corruptions, without requiring data augmentation. Further cumulative gains in robustness against this array of distortions can be obtained by appropriately combining TEXP with data augmentation techniques. The code for all our experiments is available at https://github.com/bhagyapuranik/texp_for_robustness.

1 Introduction

Standard training of deep neural networks is well known to lack robustness against a variety of distortions, including noise, distribution shifts (Hendrycks and Dietterich, 2018; Dodge and Karam, 2017), and adversarial attacks (Szegedy et al., 2014; Goodfellow et al., 2015; Carlini and Wagner, 2017). The most common approach to improving robustness relies on performing data augmentation. For example, adversarial training (Madry et al., 2018), which augments the training data with generated adversarial examples (corresponding to the current realization of the network parameters), is one of the most effective adversarial defenses against adversarial attacks. In addition, different types of data augmentation have also been shown to effectively improve robustness against natural corruptions (Cubuk et al., 2019; Hendrycks et al., 2020; Qin et al., 2023).

In this paper, in a manner that is *complementary* to learning with data augmentation, we propose and explore a strategy for enhancing robustness based on detection and estimation theoretic concepts, motivated by their success in fields such as wireless communication systems. In communication theory, the receiver tries to match the incoming signal against a number of possible signal templates, each corresponding to a different message. For signaling in Gaussian noise, correlating against these signal templates, often called matched filtering, maximizes the signal-to-noise ratio, and the posterior probability of each possible transmitted signal is obtained by feeding suitably scaled matched filter outputs to a softmax. We propose to apply these ideas to enhance the *signal-to-noise ratio* in a neural network layer (our exploration here focuses on the first layer) via neuronal competition during learning and inference.

Unlike in communication systems, we do not have a known set of messages and corresponding transmitted symbols. Rather, we seek to learn layer weights which are well matched to the set of incoming patterns, so that for each strong input, a fraction of neurons fire strongly. We accomplish this by adding tilted exponential objective for the layer, which can be interpreted as maximum likelihood estimation of *matched filter* signal templates under a Gaussian model for "data noise" (see Sec. 3). For inference, we replace batch norm by a tilted softmax, interpretable as computation of

Proceedings of the 27th International Conference on Artificial Intelligence and Statistics (AISTATS) 2024, Valencia, Spain. PMLR: Volume 238.

posterior probabilities for competing signal templates represented by the neurons. Our framework allows us to vary the amount of *data noise* during training (smaller if training with clean data) and during inference (bigger to provide robustness against out of distribution "noise"). We term a layer designed in this way (see Sec. 4) as a tilted exponential (TEXP) layer.

We provide geometric insight into TEXP training for simplified models, and then demonstrate enhancements in robustness for CNNs operating on standard image datasets. Experiments on CIFAR-10 (Krizhevsky et al., 2009) show that replacing the first layer of a VGG-16 (Simonyan and Zisserman, 2014) network by a TEXP layer yields increased robustness against noise, other common corruptions and mild adversarial perturbations *without requiring data augmentation*. Additional performance gains are obtained by supplementing TEXP with adversarial training and other data augmentation techniques such as AugMix (Hendrycks et al., 2020), RandAugment (Cubuk et al., 2020) and AutoAugment (Cubuk et al., 2019). We show that the TEXP approach generalizes to other network architectures and datasets through promising preliminary experiments on CIFAR-100 (using Wide-ResNet-28-10 as backbone) and ImageNet (ResNet-50 backbone).

2 Related work

Disparity between the data observed during training and testing phases is a common phenomenon, highlighting the significance of robustness in generalizing to out-of-distribution (OOD) samples. The prevailing approach to address this challenge is to use various forms of OOD data augmentation (Zhang et al., 2018; Cubuk et al., 2019, 2020; Schneider et al., 2020; Hendrycks et al., 2020; Calian et al., 2022; Kireev et al., 2022; Qin et al., 2022), often combined with techniques such as proxy tasks or consistency regularization. For example, Augmix (Hendrycks et al., 2020) enriches training images by incorporating a composition of randomly sampled augmentations to generate a diverse set of augmented images, supplemented by a consistency loss function aimed at encouraging DNN outputs to react smoothly to augmentations. Such consistency regularization for data augmentation has shown promise in several other works as well (Tack et al., 2022; Huang et al., 2022).

A complementary set of works (Gilmer et al., 2019; Yi et al., 2021; Yin et al., 2019; Qin et al., 2023) indicate that adversarial training (typically with small perturbation budgets) can also lead to improvements in OOD robustness. However, finding techniques that work well for various kinds of OOD corruptions, particularly without heavy data augmentation, remains challenging. For example, Yin et al. (2019) find that adversarial training and Gaussian noise augmentation improve robustness against certain corruptions like other types of noise and blurs while degrading the performance under *low frequency* corruptions like fog and contrast. They argue that a diverse set of augmentations may be required to combat such trade-offs. Our TEXP method shows promise in achieving broad spectrum robustness without data augmentations, while also combining well with strategies such as AugMix and adversarial training.

Our approach of adding a cost based on layer activations is motivated by the recent work (Cekic et al., 2022), which argues that targeting sparse, strong activations at early network layers can improve robustness. Cekic et al. (2022) employs Hebbian/anti-Hebbian (HaH) training at the layers, in which the most active neurons for an input are promoted towards the input ("fire together, wire together"), while neurons which are less active are demoted away from the input, and uses divisive normalization (enabling smaller outputs to be attenuated by larger outputs) instead of batch norm for inference. In contrast to neuroscientific motivation in HaH, our TEXP training and inference approach is derived from communication-theoretic foundations. Our approach also promotes neuronal competition in both training and inference, and results in sparse, strong activations. However, our framework leads to smoother objective functions, does not require sorting in either training or inference, and our best schemes substantially outperform Cekic et al. (2022).

Exponential tilting is well-known in statistics for rejection sampling, rare-event simulation, saddle-point approximation (Butler, 2007), and importance sampling (Siegmund, 1976). It is also at the heart of Chernoff bounds (Dembo and Zeitouni, 2009), as well as analyzing atypical events in information theory (Beirami et al., 2018), and has appeared as a smoothing method to maximum in optimization literature (Kort and Bertsekas, 1972; Pee and Royset, 2011; Liu and Theodorou, 2019). These concepts have motivated prior work on tilted exponentials applied to the *training objective function,* which has been demonstrated to yield fairness and robustness benefits in a multitude of machine learning problems (Li et al., 2021, 2023). Unlike this prior work on exponential tilting, which is motivated by connections to Chernoff bounds, large deviations and typicality, our proposal of TEXP objective is motivated by maximum likelihood estimation of signal templates, and we apply exponential tilting to layer activations.

3 Learning matched filters with TEXP

We provide here a communication-theoretic motivation for training and inference in a TEXP layer, and provide insight into why it produces sparse, strong activations, which in turn provide robustness. We also provide a

geometric illustration of the neuronal weights obtained with TEXP training via simplified models.

A classical model in communication theory is to consider the received signal as one of M possible transmitted messages, corrupted by white Gaussian noise. Under hypothesis H_i, the received signal is modeled as

$$H_i : \mathbf{x} = \mathbf{w}_i + \mathbf{n} \tag{1}$$

where $\{\mathbf{w}_i\}_{i\in[M]}$ ($[M] := \{1, \ldots, M\}$) are the M possible known messages, and $\mathbf{n}$ is white Gaussian noise with variance ν^2 per dimension.

In this setting, it is well known (Madhow, 2008) that sufficient statistics for optimal reception are obtained by correlating the received signal against each of the possible transmitted signals. That is, the optimal receiver computes the inner products $\mathbf{x}^T\mathbf{w}_i$, $i \in [M]$, to make its decisions. These correlation operations are often termed *matched filtering,* since they try to match the received signal against one of M possible templates.

The proposed tilted exponential (TEXP) approach for robustness is based on fitting the model (1) to the input $\mathbf{x}$ to a layer in a neural network (the experimental results presented in this paper focus exclusively on the first layer of a convolutional neural network, but in principle, the concepts could be applied to any layer of a neural network). For a given layer of a neural network with M neurons (or M output channels or filters in case of a CNN), we interpret the neuronal weights $\mathcal{W} = \{\mathbf{w}_i\}_{i\in[M]}$ as signal templates corresponding to M possible hypotheses regarding the input $\mathbf{x}$, with the ith neuron producing the matched filter output $\mathbf{x}^T\mathbf{w}_i$, $i \in [M]$. However, unlike in a communication system, we do not know the set of possible "messages" *a priori.* We must therefore learn the "matched filters" $\mathcal{W}$ based on input training samples during training, and then perform inference based on these learnt templates. Naturally, we do not expect such a model to be accurate, but fitting it to data provides an approach for learning neural weights such that, for each input, it is likely that there is a subset of neurons well matched to it. The parameter ν^2 may be viewed as "data noise," acknowledging that the input $\mathbf{x}$ may not fit any of the templates we learn.

Likelihood function. For the model (1), the conditional density of $\mathbf{x}$ under hypothesis H_i is $p(\mathbf{x}|H_i) = N(\mathbf{x}|\mathbf{w}_i, \nu^2\mathbf{I})$, where $N(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is the Gaussian density with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. However, a more convenient representation for maximum likelihood estimation is to take its Radon-Nikodym derivative with respect to the conditional density $N(\mathbf{x}|\mathbf{0}, \nu^2\mathbf{I})$ for a "noise-only" dummy hypothesis, to obtain the conditional likelihoods:

$$L_{\mathcal{W}}(\mathbf{x}|H_i) = \frac{N(\mathbf{x}|\mathbf{w}_i, \nu^2\mathbf{I})}{N(\mathbf{x}|\mathbf{0}, \nu^2\mathbf{I})} = \exp\left(\frac{1}{\nu^2}(\mathbf{x}^T\mathbf{w}_i - \|\mathbf{w}_i\|^2/2)\right), \tag{2}$$

for $i \in [M]$, where we now view $\mathcal{W}$ as parameters to be learnt during training. We see that these conditional likelihoods depend on the input $\mathbf{x}$ only through the matched filter outputs $\mathbf{x}^T\mathbf{w}_i$, $i \in [M]$, which means that these are sufficient statistics.

Assuming that all templates have equal energy, we can drop the $\|\mathbf{w}_i\|^2/2$ terms from (2) to obtain the simplified expression, for all $i \in [M]$:

$$L_{\mathcal{W}}(\mathbf{x}|H_i) = \exp\left(\frac{1}{\nu^2}\mathbf{x}^T\mathbf{w}_i\right). \tag{3}$$

Averaging over the conditional likelihoods (3), the likelihood of $\mathbf{x}$ is obtained as a sum of tilted exponentials:

$$L_{\mathcal{W}}(\mathbf{x}) = \frac{1}{M}\sum_{i=1}^{M}\exp\left(\frac{1}{\nu^2}\mathbf{x}^T\mathbf{w}_i\right) = \frac{1}{M}\sum_{i=1}^{M}\exp\left(ta_i\right), \tag{4}$$

where $t = \frac{1}{\nu^2} > 0$ is the tilt parameter and $a_i = \mathbf{x}^T\mathbf{w}_i$ is the activation (or matched filter output) produced by the i-th neuron.

TEXP training objective. Maximum likelihood estimation of signal templates using a collection of independently drawn data points is accomplished by maximizing the sum of log-likelihoods. This corresponds to the following tilted exponential objective function:

$$\mathcal{L}_{\text{TEXP}}(\mathbf{x}) = \log L_{\mathcal{W}}(\mathbf{x}) = \log\frac{1}{M}\sum_{i=1}^{M}\exp\left(ta_i\right). \tag{5}$$

Implicit normalization of templates. While different transmitted signals in a communication system might have different energies, we wish to enforce fair competition across templates by normalizing each to unit norm. This can be accomplished without explicit optimization constraints via implicit normalization of activations, by redefining as: $a_i = \frac{\mathbf{x}^T\mathbf{w}_i}{\|\mathbf{w}_i\|_2}$, $i \in [M]$.

Notation (softmax). Before computing the gradient of the TEXP objective, we recall the standard notation $\sigma(\mathbf{u}) = (\sigma_1(\mathbf{u}), ..., \sigma_M(\mathbf{u}))^T$ for a softmax operating on a vector $\mathbf{u} = (u_1, ..., u_M)^T$, with

$$\sigma_i(\mathbf{u}) = \frac{e^{u_i}}{\sum_{j=1}^{M} e^{u_j}}\ ,\ i = 1, ..., M \tag{6}$$

TEXP gradient. The gradient of the objective function (5) is obtained as

$$\nabla_{\mathcal{W}}\mathcal{L}_{\text{TEXP}} = t\sum_{i=1}^{M}\frac{e^{ta_i}}{\sum_{j=1}^{M} e^{ta_j}}\nabla_{\mathcal{W}}a_i = t\sum_{i=1}^{M}\sigma_i(t\mathbf{a})\nabla_{\mathcal{W}}a_i, \tag{7}$$

Since larger activations are weighted more via the tilted softmax, gradient ascent corresponds to increasing larger activations further.

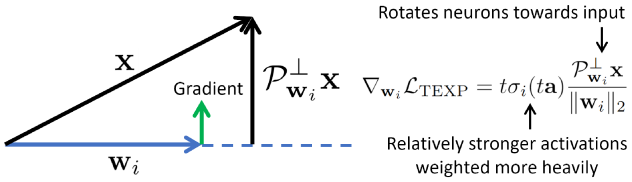


Figure 1: TEXP gradient ascent rotates the neuron towards the input, with larger activations weighted more via a tilted softmax.

Accounting for implicit normalization, we obtain the gradient

$$\nabla_{\mathbf{w}_i} a_i = \nabla_{\mathbf{w}_i} \frac{\mathbf{x}^T\mathbf{w}_i}{\|\mathbf{w}_i\|_2} = \frac{\mathcal{P}^{\perp}_{\mathbf{w}_i}\mathbf{x}}{\|\mathbf{w}_i\|_2}, \text{ where,}$$

$$\mathcal{P}^{\perp}_{\mathbf{w}_i}\mathbf{x} = \mathbf{x} - \left(\frac{\mathbf{x}^T\mathbf{w}_i}{\|\mathbf{w}_i\|_2}\right)\frac{\mathbf{w}_i}{\|\mathbf{w}_i\|_2}$$

is the projection of the input $\mathbf{x}$ orthogonal to the one-dimensional subspace spanned by $\mathbf{w}_i$ (Fig. 1). Note that, for $j \neq i$, $\nabla_{\mathbf{w}_j} a_i = 0$.

We can now write $\nabla_{\mathcal{W}}\mathcal{L}_{\text{TEXP}} = (\nabla_{\mathbf{w}_1}\mathcal{L}_{\text{TEXP}}, \ldots, \nabla_{\mathbf{w}_M}\mathcal{L}_{\text{TEXP}})$, where

$$\nabla_{\mathbf{w}_i}\mathcal{L}_{\text{TEXP}} = t\sigma_i(t\mathbf{a})\frac{\mathcal{P}^{\perp}_{\mathbf{w}_i}\mathbf{x}}{\|\mathbf{w}_i\|_2}. \tag{8}$$

We now see in explicit form that a TEXP gradient update rotates each neuron to align more closely with the input (see Fig. 1), with the softmax weighting favoring templates that are better aligned, so that large activations are made even larger.

Balanced TEXP objective function. Additional competition among the signal templates seeking to fit an input can be created by imposing a *balance* constraint in which the mean of the signal templates is set to zero. That is, we replace $\mathbf{w}_i$ by $\mathbf{w}_i - \bar{\mathbf{w}}$, for $i \in [M]$, where $\bar{\mathbf{w}} = (1/M)\sum_{i=1}^{M}\mathbf{w}_i$. This yields a variant of (5) which we term a *balanced* tilted exponential objective function:

$$\mathcal{L}^{\text{bal}}_{\text{TEXP}}(\mathbf{x}) = \log\frac{1}{M}\sum_{i=1}^{M}\exp\left(t(a_i - \bar{a})\right),$$

where $\bar{a} = (1/M)\sum_{i=1}^{M} a_i$ is the mean activation of all neurons. The corresponding gradient components are given by

$$\nabla_{\mathbf{w}_i}\mathcal{L}^{\text{bal}}_{\text{TEXP}} = t\left(\sigma_i(t\mathbf{a}) - 1/M\right)\frac{\mathcal{P}^{\perp}_{\mathbf{w}_i}\mathbf{x}}{\|\mathbf{w}_i\|_2} \tag{9}$$

Now, in addition to making large activations larger by rotating towards the input, we make small activations smaller (i.e., such that tilted softmax is smaller than $1/M$) by by rotating the corresponding template *away* from the input (for geometric insights, see Sec 3.2).

TEXP inference. Once we learn the estimates of the signal templates $\mathcal{W}$, inference based on a data point $\mathbf{x}$ consists of computing the posterior probability of each hypothesis (this is termed "soft decisions" in communication systems). For hypothesis H_i, this posterior probability is given by the softmax:

$$p_i(\mathbf{x}) = \frac{L_{\mathcal{W}}(\mathbf{x}|H_i)P(H_i)}{\sum_{j=1}^{M} L_{\mathcal{W}}(\mathbf{x}|H_j)P(H_j)} = \sigma_i\left(\mathbf{a}/\nu^2\right) = \sigma_i(t\mathbf{a}) \tag{10}$$

setting $t = \frac{1}{\nu^2}$.

Different tilt parameters for training and inference. The value of ν^2 used during inference using (10) may be different from that for training as in (5). In particular, we may use a smaller value of ν^2 (higher t) during training, where we might be learning from clean data, or from data that we have perturbed in a controlled manner. On the other hand, during inference, we may use a higher value of ν^2 (lower t) in order to accommodate data noise due to a variety of distortions that were not present during training. Note that TEXP inference (10) is unaffected by whether or not the signal templates are balanced, since balancing corresponds to subtracting the same constant from each activation.

3.1 Why TEXP is expected to reduce sensitivity to perturbations.

TEXP training pushes activations from different neurons apart, nudging different neurons to align with different signal templates. This combines well with TEXP inference: the softmax nonlinearity enables large activations, corresponding to neurons well-aligned with the input, to suppress smaller activations, and reduce sensitivity to perturbations. To see this, consider a layer with only two neurons, with activations a_1 and a_2. Defining $\Delta a = a_1 - a_2$ as the difference in activations, the softmax outputs reduce to sigmoids:

$$z_1 = \sigma_1(t\mathbf{a}) = f(-t\Delta a), \quad z_2 = \sigma_2(t\mathbf{a}) = f(t\Delta a)$$

where $f(x) = 1/(1+e^{-x})$ is the standard single-argument sigmoid function. Since $f(x) \to 0$ as $x \to -\infty$, the derivative of the sigmoid, $f'(x) = f(x)f(-x)$, is small for large $|x|$. Thus, as we increase the separation between the activations (by increasing $|\Delta a|$), the *sensitivity* of the softmax output to perturbations decreases. This is in contrast to the ReLU nonlinearity, where perturbations can ride on top of activations in the linear region.

Remark 1. Even in this simplified setting of two neurons, note that the TEXP inference layer is different from a classical sigmoid nonlinearity. The TEXP inference output depends on the sigmoid of the *difference* in activations, instead of on individual activations as in a classical setting.

Remark 2. As the tilt parameter increases, the sensitivity to perturbations decreases, but this may come at the cost of excessive loss of information due to suppression of weaker activations.

Remark 3. The suppression of small activations by larger ones via softmax leads to a sparse code with a small fraction of strong activations. It is beneficial to further threshold the softmax layer output to zero out small entries, further increasing sparsity and reducing the effective number of dimensions available for perturbations to propagate up.

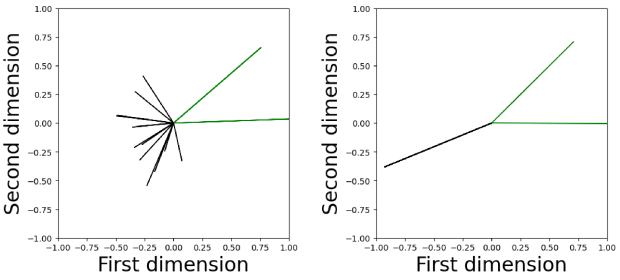
3.2 Geometric insight into TEXP learning.

While our evaluations in Sec. 5 focus on TEXP as a supplement to empirical risk minimization for supervised learning, to obtain geometric insight, we consider unsupervised TEXP training on a single-layer network for two simplified data models, each corresponding to a 2-dimensional *signal subspace* embedded in an ambient dimension $d \gg 2$. We apply TEXP training on M randomly initialized neurons $\{\mathbf{w}_i\}_{i\in[M]}$ (where $M \gg 2$), and ask if we learn neurons aligned with the *important* directions in the input distribution. Since the neurons are randomly initialized, we can assume, without loss of generality, that the first 2 elements of the standard basis, $\mathbf{e}_1 = [1, 0, \ldots, 0]$ and $\mathbf{e}_2 = [0, 1, 0, \ldots, 0]$, span the 2-dimensional signal subspace.

Model 1: The input is drawn from a two-component Gaussian mixture, corresponding to one of two equiprobable signals, $\mathbf{s}_1$ and $\mathbf{s}_2$, corrupted by white Gaussian noise of variance σ^2 per dimension. The input distribution is therefore given by

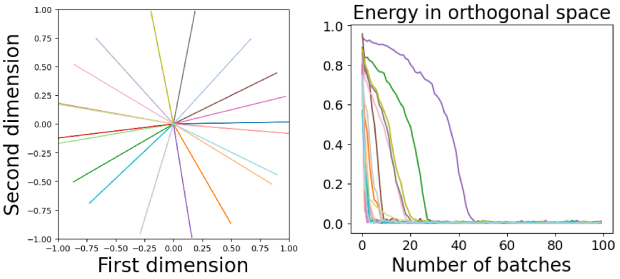
$$p(\mathbf{x}) = \frac{1}{2}\mathcal{N}(\mathbf{x}|\mathbf{s}_1, \sigma^2\mathbf{I}) + \frac{1}{2}\mathcal{N}(\mathbf{x}|\mathbf{s}_2, \sigma^2\mathbf{I}) \qquad (11)$$

where $\mathbf{x} \in \mathcal{R}^d$. This is exactly aligned with our communication-theoretic formulation with two hypotheses. We hope that we learn one or more neurons that align with each of the signal directions $\mathbf{s}_1$ and $\mathbf{s}_2$, and that outputs corresponding to the remaining non-useful neurons are small enough that they can be suppressed via TEXP inference. Additionally, when we employ the balanced TEXP objective, the inactive neurons are expected to be rotated away from the signal directions. We consider an example with signals $\mathbf{s}_1 = [1, 0, \ldots, 0]$, $\mathbf{s}_2 = [1/\sqrt{2}, 1/\sqrt{2}, 0, \ldots, 0]$, $d = 10$, $M = 20$, and we plot in Fig. 2(a) the projections, of the learnt neurons, onto the signal space. As expected, TEXP leads to some *useful* neurons (shown in green) that align with the signal directions, while the inner products between the remaining *non-useful* or *spurious* neurons (shown in black) and the signals are either negative or are small positive numbers (and hence would be significantly attenuated by the softmax in TEXP inference relative to the activations of the useful neurons). The projections of the non-useful neurons into the signal space are of various lengths; that is, they may have substantial energy orthogonal to the signal space. With the *balanced* TEXP objective, on the other hand, the non-useful neurons (in black, Fig. 2(b)) are rotated away from the signal directions, so that their inner products with signals are negative. Further, the energy of the non-useful neurons is also concentrated in the signal space.



(a) TEXP (b) Balanced TEXP

Figure 2: Projection of learnt neurons in signal subspace for model 1



(a) (b)

Figure 3: Convergence of neurons for model 2 and their respective energies in the orthogonal space.

Model 2: The input is a zero mean Gaussian random vector with density $p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\mathbf{0}, \mathbf{C})$, with eigenvectors taken to be aligned with the standard basis without loss of generality, and the first two dominant eigen-directions defining the signal subspace:

$$\mathbf{C} = \text{diag}\left(A_1^2 + \sigma^2, A_2^2 + \sigma^2, \sigma^2, ..., \sigma^2\right) \qquad (12)$$

where A_1^2, A_2^2 are *signal powers* in the dominant eigen-directions, and σ^2 is the ambient noise variance per dimension. We can rewrite the model as a random Gaussian signal in a two-dimensional subspace corrupted by white Gaussian noise of variance σ^2 per dimension: $\mathbf{x} = \mathbf{s} + \mathbf{N}$, where $\mathbf{s} = A_1 Z_1 \mathbf{e}_1 + A_2 Z_2 \mathbf{e}_2$ with Z_1, Z_2 i.i.d. $\mathcal{N}(0, 1)$, and $\mathbf{N} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$ independent of Z_1, Z_2. Since (Z_1, Z_2) takes a continuum of values in two dimensions, we would need a continuum of signal hypotheses to fit this into our communication-theoretic formulation. Thus, given $M \gg 2$ neurons, we expect TEXP training to utilize all available neurons to obtain a sparse code for the random signal based on appropriately quantizing the continuum of directions that $\mathbf{s}$ can take in the two-dimensional space. Fig. 3(a), where we plot the projections of all $M = 20$ neurons (in distinct colors) onto the signal space, shows that the energy of the learnt neurons is indeed concentrated

in the signal subspace, with more neurons aligned more closely along the first dimension (since $A_1 > A_2$; we set $A_1 = 3$, $A_2 = 2$). Fig. 3(b) plots the orthogonal components of the energies of each of these neurons (in respective colors) on the Y-axis, and shows how the energy orthogonal to the signal subspace dies down quickly during training.

The contrasting behavior with these simplified data models indicates how TEXP learning adapts to the richness of the signal subspace to create a sparse code. More insights on the sparsity induced by TEXP are deferred to Appendix A.

4 TEXP as a neural network layer

We now translate these ideas to training a convolutional layer in a CNN (the description specializes in a straightforward manner to a fully connected layer).

TEXP inference layer. We replace a conventional ReLU and batchnorm in the first layer of a neural network by a *tilted softmax* and *thresholding layer*, and supplement the end-to-end training objective with the TEXP objective for learning matched filters. Our exposition focuses on replacing the first layer by a TEXP layer, but in principle this could be applied to multiple layers.

Convolution with implicit normalization: For a CNN layer with input $\mathbf{x}$, we denote the parameters by $\mathcal{W} = \{\mathbf{w}_i\}_{i\in[M]}$ (where M denotes the number of output channels or the number of filters), with each filter $\mathbf{w}_i$ being a $k \times k$ kernel with c_{in} input channels. Thus the dimension of $\mathbf{w}_i$ is $D = k \times k \times c_{\text{in}}$. Assuming k is odd, consider $\mathbf{x}(l)$ to be a D-dimensional patch of the input, centered around the spatial location l, where $l \in [L]$ and L is the total number of such patches which are convolved with the filters, which depends on the dimensions of the input and the striding and padding. For example, for CIFAR-10 images fed to a VGG-16 model, the first convolution block consists of $M = 64$ filters, each a 3×3 kernel with stride and padding of 1. Thus, we have $L = 32 \times 32 = 1024$ spatial locations and corresponding input patches. Similarly to Cekic et al. (2022), we implicitly normalize the convolution filter weights to unit ℓ_2 norm, leading to the following convolution output produced at spatial location l due to the i-th filter, computed as a tensor inner product as follows:

$$y_i(l) = \frac{(\mathbf{x}(l))^T \mathbf{w}_i}{\|\mathbf{w}_i\|_2}. \tag{13}$$

Tilted softmax: Post the convolution, we pass the convolution outputs at each location l through a Tilted Softmax (TS) to obtain *posterior probabilities*

$$p_i(l) = \frac{\exp(t_{\text{inf}} y_i(l))}{\sum_{j=1}^{M} \exp(t_{\text{inf}} y_j(l))} = \sigma_i(t_{\text{inf}} \mathbf{y}(l)),$$

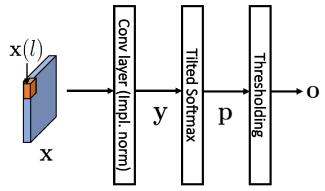


Figure 4: The illustration of a TEXP layer.

where $\mathbf{y}(l) = \{y_i(l), i = 1, 2, \ldots, M\}$, t_{inf} is the tilt parameter. This enforces competition across filters at each location l.

Thresholding: A filter-specific data-adaptive thresholding is performed to obtain TEXP layer outputs:

$$o_i(l) = \begin{cases} p_i(l) & \text{if } p_i(l) \geq \tau_i \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

This idea of adaptive thresholding is borrowed from Cekic et al. (2022), where τ_i was set such that a certain fraction (e.g., 10 or 20%) of the outputs are activated for each filter for any given input. However, we avoid the sorting required for the latter (thus significantly reducing complexity) by setting $\tau_i = m_i + c\sigma_i$, where $m_i = \frac{1}{L}\sum_{l\in[L]} o_i(l)$ is the mean of the softmax outputs for filter i, σ_i is corresponding the standard deviation, and c is a hyperparameter (we set $c = 0.5$ in our experiments).

TEXP objective. The TEXP objective for a given layer contributed by each training data point is:

$$\mathcal{L}_{\text{TEXP}} = \frac{1}{L}\sum_{l=1}^{L} \frac{1}{t} \log\left(\frac{1}{M}\sum_{i=1}^{M} \exp(t y_i(l))\right) \tag{15}$$

where t is the training tilt parameter, chosen to be at least as large as the inference tilt t_{inf} when learning with clean data. The overall cost function takes the form

$$\mathcal{L} = \mathcal{L}_{\text{CE}} - \alpha \mathcal{L}_{\text{TEXP}},$$

where $\mathcal{L}_{\text{CE}}$ is the standard cross-entropy loss and $\alpha > 0$ is a hyperparameter that determines the relative importance of the TEXP objective compared to the discriminative cross-entropy loss.

5 Experimental setup

We first present details on the baseline and TEXP models employed in the experimental evaluation, the training mechanism and the test metrics. We then report on experiments demonstrating enhancements of robustness for supervised learning of CNNs, replacing the first DNN layer of a baseline architecture with a TEXP layer. Our primary focus is on the CIFAR-10 standard and corruption datasets with the VGG-16 model as the baseline architecture, where we expect significant gains in robustness to noise and other common corruptions in the TEXP models. The code for all our experiments is available at https://github.com/bhagyapuranik/texp_for_robustness.

Baselines. We obtain baseline VGG-16 models (with implicit weight normalization), with standard training (which is not expected to be robust), with OOD data augmentation techniques such as AugMix, RandAugment, AutoAugment, and with PGD-based adversarial training (Madry et al., 2018) with ℓ_∞ perturbations of budget $\epsilon = 2/255$ (which is expected to be robust against a number of other corruptions as well (Yi et al., 2021)). The HaH model (Cekic et al., 2022) is also used as a baseline for OOD robustness. Like our approach, it supplements training with layer-wise costs. The HaH model modifies 6 layers, but as we shall see, the TEXP approach with a single layer outperforms it.

Our models. We modify the first layer of the VGG-16 to a TEXP layer and refer to the model as TEXP-VGG-16. We simplify hyperparameter search for tilt parameters via the following scaling arguments. In view of the implicit normalization of the filters, activations scale with the ℓ_2 norm of the input $\|\mathbf{x}\|_2$ to the filter, so that the tilt parameter should be chosen to compensate for this scaling. Making the simplifying assumption that input components are uncorrelated, the energies of the input components add up, and we may assume that $\|\mathbf{x}\|_2$ scales as $\sqrt{D}$, where D is the dimension of the filter tensor (e.g., for the input of VGG-16, $D = 3 \times 3 \times 3 = 27$). Instead of an extensive hyperparameter search, we set $t_{\text{inf}} = 1/\sqrt{D} = 0.192$, which results in relatively "soft" decisions at the softmax output. For the clean training data, we set $t = 10/\sqrt{D} = 1.92$, so TEXP training makes "harder" decisions in favor of winners when learning signal templates. The relative weight of the TEXP objective (15) in comparison to the cross-entropy loss is set to $\alpha = 0.001$, chosen so that the magnitude of the weighted TEXP objective is smaller than the cross-entropy loss. We use the same parameters when supplementing TEXP with data augmentation techniques and adversarial training with ℓ_∞ perturbations of budget $\epsilon = 2/255$ (better performance may be obtained by further fine-tuning TEXP parameters for each setting). We provide a detailed study on the sensitivity of the TEXP models to the different TEXP parameters in Appendix B. We also report on an alternate, computationally intensive variant of TEXP, in Appendix C.

Training. For all VGG-16 based models, we employ the ADAM optimizer (Kingma and Ba, 2014) with a multi-step learning rate, beginning with 0.001, and decreasing by a factor of 10 at epochs 60 and 80. We train the models for 100 epochs.

Evaluation metrics. We evaluate over 19 different common corruptions on the CIFAR-10-C (Hendrycks and Dietterich, 2018) dataset. We report the test accuracies for minimum (worst-case) and average over all the corruptions, for both the entire dataset comprising of 5 different severity levels, and also on specifically the corruptions of the highest severity.We also separately report on the corrupted data formed by the addition of Gaussian noise with standard deviation $\nu = 0.1$ (since our TEXP formulation was motivated by hypothesis testing with Gaussian noise, we expect enhanced robustness to Gaussian noise). Finally, we explore whether TEXP (even without adversarial training) enhances resilience against mild adversarial attacks, by evaluating all trained models on different ℓ_p attacks such as ℓ_1 with budget $\epsilon = 3$, ℓ_2 with $\epsilon = 0.25$ and ℓ_∞ with $\epsilon = 2/255$ respectively. We use AutoAttack (Croce and Hein, 2020), suggested by RobustBench (Croce et al., 2020), which is parameter-free and consists of a suite of different attacks (in particular, we employ the *standard* version composed of APGD-CE, APGD-T, FAB-T and Square attacks).

TEXP is intended to produce sparse, strong activations; we report on sparsity measures for TEXP layer outputs in Appendix D.

Broader applicability. We illustrate the applicability of TEXP to different architectures and larger datasets via preliminary experiments on the CIFAR-100 dataset using Wide-ResNet-28-10 (Zagoruyko and Komodakis, 2016) baseline and on ImageNet-1K (Russakovsky et al., 2015) with ResNet50 baseline. For TEXP-WideResNet-28-10, we set the TEXP parameters as $t_{\text{inf}} = 1/\sqrt{D} = 0.192$, $t = 4t_{\text{inf}} = 0.768$ and $\alpha = 0.0005$. For TEXP-ResNet-50, we set the TEXP parameters as $t_{\text{inf}} = 8/\sqrt{D} = 0.656$, $t = 10t_{\text{inf}} = 0.656$ and $\alpha = 0.01$. For both these ResNet family backbones, we employ SGD optimizer with momentum (0.9), initial learning rate of 0.1, and weight decay. The ResNet-50 models are trained for 90 epochs, with a 10× learning rate reduction every 30 epochs. The WideResNet models are trained for 200 epochs, with 5× learning rate reduction every 60 epochs.

6 Experimental results

Improvement in robustness against OOD corruptions. In Table 1, we report the enhanced *general-purpose* robustness of the TEXP approach on the CIFAR-10 dataset. The reported numbers are test accuracies averaged across five runs, along with the standard error. Please refer to Appendix E for details on the computation of the standard errors. Focusing on the robustness to common corruptions, we can observe that TEXP provides significant gains in robustness to noise and other out-of-distribution (OOD) corruptions (both at the highest severity level and all levels) in comparison to standard VGG and HaH baselines. We benefit substantially against the worst corruption (i.e., the one with minimum accuracy among all corruptions). TEXP combined with data augmentations

Model	Clean	OOD corruptions			Adversarial perturbations		
		Noise $\nu = 0.1$	Min/Avg corruptions	Min/Avg severity level:5	Autoattack ℓ_1 adv, $\epsilon = 3$	Autoattack ℓ_2 adv, $\epsilon = 0.25$	Autoattack ℓ_∞ adv, $\epsilon = 2/255$
VGG-16	92.26 ± 0.04	24.80 ± 1.24	$46.86 \pm 1.26/72.28 \pm 0.26$	$19.56 \pm 0.73/54.70 \pm 0.40$	10.14 ± 0.22	13.34 ± 0.14	10.30 ± 0.21
HaH (Cekic et al. (2022))	87.72 ± 0.15	62.76 ± 0.40	$59.56 \pm 0.42/77.02 \pm 0.21$	$49.06 \pm 0.88/67.80 \pm 0.27$	29.98 ± 0.45	26.30 ± 0.52	20.04 ± 0.38
TEXP-VGG-16	88.28 ± 0.12	75.14 ± 0.20	$73.68 \pm 0.22/80.40 \pm 0.07$	$52.38 \pm 0.81/72.56 \pm 0.14$	46.48 ± 0.95	50.90 ± 0.16	41.50 ± 0.21
VGG-16 + AugMix	92.98 ± 0.06	62.92 ± 0.74	$65.12 \pm 0.35/83.58 \pm 0.09$	$42.12 \pm 0.79/74.00 \pm 0.16$	17.88 ± 0.26	18.16 ± 0.15	13.60 ± 0.17
TEXP-VGG-16 + AugMix	88.84 ± 0.21	78.90 ± 0.04	$77.28 \pm 0.20/83.54 \pm 0.05$	$62.94 \pm 0.53/78.30 \pm 0.07$	48.68 ± 0.52	52.20 ± 0.23	42.52 ± 0.20
VGG-16 + RandAug	93.32 ± 0.11	43.32 ± 0.72	$63.24 \pm 0.45/80.68 \pm 0.17$	$39.98 \pm 1.01/66.96 \pm 0.30$	19.76 ± 0.08	18.38 ± 0.47	14.30 ± 0.37
TEXP-VGG-16 + RandAug	89.90 ± 0.08	74.26 ± 0.07	$75.48 \pm 0.09/82.86 \pm 0.02$	$57.52 \pm 0.19/75.78 \pm 0.07$	50.16 ± 0.24	50.82 ± 0.24	40.02 ± 0.34
VGG-16 + AutoAug	93.50 ± 0.03	46.54 ± 0.54	$59.84 \pm 0.52/81.58 \pm 0.14$	$37.08 \pm 0.23/70.66 \pm 0.18$	15.66 ± 0.19	13.50 ± 0.23	9.78 ± 0.20
TEXP-VGG-16 + AutoAug	90.06 ± 0.10	72.66 ± 0.46	$71.98 \pm 0.24/82.58 \pm 0.12$	$54.14 \pm 0.89/75.50 \pm 0.18$	47.62 ± 0.16	46.96 ± 0.31	35.00 ± 0.32
VGG-16 + Adv Tr	88.04 ± 0.12	78.78 ± 0.45	$50.52 \pm 0.66/79.44 \pm 0.12$	$17.60 \pm 0.39/70.64 \pm 0.13$	51.26 ± 0.71	72.60 ± 0.23	72.82 ± 0.23
TEXP-VGG-16 + Adv Tr	86.38 ± 0.07	81.08 ± 0.28	$67.72 \pm 0.73/80.38 \pm 0.14$	$37.08 \pm 0.85/74.02 \pm 0.22$	64.50 ± 0.45	71.02 ± 0.40	66.76 ± 0.29

Table 1: Enhanced robustness to corruptions and mild adversarial attacks under VGG-16 based TEXP models on CIFAR-10 clean and corruptions datasets. All numbers reported are average test accuracies ± standard error.

Corruptions →	Noise				Weather					Blur					Digital				
Models ↓	Gauss.	Shot	Speck.	Imp.	Snow	Frost	Fog	Brig.	Spat.	Defoc.	Gauss.	Glass	Motion	Zoom	Cont.	Elas.	Pixel.	JPEG	Satur.
VGG-16	24.3	31.8	38.4	19.1	73.3	62.0	63.8	87.9	67.3	50.8	39.8	47.6	60.0	61.5	19.9	75.6	54.6	77.4	82.4
HaH (Cekic et al., 2022)	61.7	61.7	59.2	46.3	73.8	72.3	62.8	83.2	76.7	64.3	58.4	53.2	65.1	68.9	76.0	74.0	60.5	79.3	79.6
TEXP-VGG-16	75.3	76.5	75.5	61.3	76.4	76.8	51.8	83.2	76.1	68.9	63.4	68.6	65.0	74.2	66.0	75.2	80.8	82.9	78.8
VGG-16 + AugMix	60.7	68.1	71.3	44.9	80.2	75.3	76.5	89.7	81.7	84.8	80.8	59.6	81.4	84.0	40.0	79.5	69.4	82.0	86.9
TEXP-VGG-16 + AugMix	78.9	79.5	79.0	67.7	78.4	79.0	62.2	83.8	78.8	81.5	79.8	72.4	77.1	82.6	75.5	78.6	83.6	83.7	81.6
VGG-16 + RandAug	44.7	53.5	57.5	40.0	78.6	72.8	71.0	90.9	85.3	63.6	52.9	61.0	67.8	71.7	48.3	79.9	56.9	81.7	88.5
TEXP-VGG-16 + RandAug	74.1	75.1	72.7	57.1	79.1	78.7	60.3	88.6	81.3	73.4	68.7	70.7	70.8	77.4	83.5	78.3	79.4	84.5	85.8
VGG-16 + AutoAug	45.7	53.1	56.7	37.1	77.2	69.8	81.1	91.9	81.1	79.1	75.2	51.8	75.2	81.1	80.0	76.5	50.4	80.2	90.2
TEXP-VGG-16 + AutoAug	72.3	72.5	70.8	53.1	76.9	76.1	62.9	88.3	77.5	76.1	72.9	65.6	72.4	79.8	86.0	76.5	77.4	84.5	86.2
VGG-16 + Adv Tr	79.8	81.1	80.3	62.7	74.3	73.3	33.2	76.8	77.7	71.1	66.8	76.0	69.1	74.9	18.3	78.4	82.6	84.8	76.6
TEXP-VGG-16 + Adv Tr	81.6	82.3	81.9	74.8	71.9	75.8	39.0	76.9	78.5	75.9	72.8	76.8	73.1	78.3	52.9	78.6	83.2	84.0	76.3

Table 2: Robustness to common corruptions of the highest severity level in the CIFAR-10-C dataset. All numbers reported are test accuracies.

and adversarial training provides even more powerful enhancements in OOD robustness, outperforming the backbones, both in terms of average over all kinds of corruptions and the minimum or worst-case among the different corruptions. We observe that TEXP augmented with AugMix provides the best OOD robustness overall. Table 2 reports the robustness of the models to each of the 19 common corruptions (in the CIFAR-10-C dataset) separately for the highest severity level of 5, and shows that TEXP models are superior in obtaining robustness against most types of corruptions, and in particular, against various noise corruptions. The robust accuracies here are reported for a single run. While vanilla adversarial training helps in robustness to noise, it deteriorates performance against contrast (Yin et al., 2019; Kireev et al., 2022; Machiraju et al., 2022). This issue is alleviated by TEXP.

Robustness against mild adversarial perturbations. Table 1 also shows that TEXP-based models enhance robustness to mild adversarial perturbations even without adversarial training. When TEXP is combined with adversarial training, its adversarial robustness generalizes significantly better than the baseline: for example, adversarial training with ℓ_∞-bounded attacks yields improved performance against (a very different kind of) ℓ_1-bounded attack.

Robustness under different architectures and datasets. The performance of the TEXP models is contrasted against baselines for the CIFAR-100 (with WideResNet-28-10) and ImageNet-1k (with ResNet-50) datasets in Table 3. We chose the TEXP parameters for these models through a mild search, settling on a combination that sacrifices clean accuracy by $3-4\%$, as we had for CIFAR-10. The results demonstrate improved OOD robustness despite minimal effort in hyperparameter tuning. We expect that more fine-grained adjustments of tilts combined with replacing multiple deeper layers with TEXP will further enhance performance, but these preliminary results do illustrate the potential gains from applying TEXP to different architectures and larger datasets.

Ablation study and connections to activation pruning. To understand the contributions of each of the components of our approach, we train and evaluate models by cumulatively adding the thresholding, tilted softmax and TEXP objective to baseline VGG. Detailed results are presented in Appendix F which show the cumulative benefits through improved robustness to both corruptions and mild adversarial attacks. An interesting connection, among works following thresholding strategies, is the work on stochastic activation pruning (SAP) (Dhillon et al., 2018), proposed as an adversar-

Model	Clean		Noise $\nu = 0.1$		Min/Avg corruptions		Min/Avg severity level:5	
	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
WRN-28-10 (CIFAR-100)	81.2	95.3	9.6	25.6	17.8/51.3	35.6/72.9	2.9/34.8	10.9/57.6
TEXP-WRN-28-10	78.4	93.6	31.4	57.3	26.0/56.7	52.8/78.7	12.4/40.9	30.1/65.0
ResNet50 (ImageNet)	75.5	92.6	55.6	79.1	24.3/38.3	41.0/58.8	3.3/17.9	8.9/33.5
TEXP-ResNet-50	72.0	90.6	62.6	84.3	26.6/41.8	45.8/62.5	3.3/21.0	9.4/37.5

Table 3: Test accuracy for WRN-28-10 based TEXP model on CIFAR-100 clean and corruptions datasets and ResNet-50 based TEXP model on ImageNet-1K clean and corruptions datasets.

ial defense and subsequently broken in Dhillon and Carlini (2020). SAP also results in sparse activations, retaining activations at each layer with probabilities proportional to their magnitude, while pruning the rest. Our deterministic thresholding scheme, used by itself, serves as a proxy for the strategy in SAP. However, the ablations in Appendix F show that the following key aspects of our approach not present in SAP are critical in enhancing robustness: (a) the TEXP objective aligns filters to match incoming patterns to promote strong activations, so that activation pruning results in less information loss, (b) TEXP inference employs a softmax nonlinearity which enhances resilience, as discussed in Section 3.

Computation cost. The average time taken to execute 1 epoch of training of a VGG-16 baseline is 5.93 sec, in comparison to 8.23 sec for TEXP-VGG-16, on an NVIDIA A100 GPU, while inference over the entire CIFAR-10 dataset takes an average of 1.13 sec for baseline and 1.12 sec for TEXP. The average training (inference) time over an NVIDIA GeForce 1080 Ti is 18.9 sec (2.79 sec) for baseline VGG and 27.5 sec (2.86 sec) for TEXP. We acknowledge that our TEXP code is not optimized for compute speed, and we believe that the training time may be improved. On the other hand, we observe that introducing TEXP layer results in little inference-time overhead to the network. In contrast to the adaptive thresholding of Cekic et al. (2022), we do not require sorting of activations to set the thresholds, significantly improving computation efficiency. Please refer to Appendix G for additional details.

7 Conclusion

Our communication-theoretic approach enhances robustness via neuronal competition in representing layer inputs during both training and inference: for a Gaussian model of data noise, the TEXP learning objective is derived as maximum likelihood estimation of signal templates, and TEXP inference as posterior probability computation. Geometric insight via unsupervised learning on simplified models illustrates how TEXP adapts to the richness of the set of possible inputs, while supplementing supervised learning in CNNs with TEXP training and inference at the input layer is shown to yield gains in OOD robustness for image datasets. Extensive experiments on a VGG model for CIFAR-10 demonstrate that, in addition to robustness gains with TEXP alone, our approach also combines well with data augmentation strategies. Preliminary results with a TEXP input layer for ResNet architectures for CIFAR-100 and ImageNet also demonstrate gains in robustness, indicating the promise of this approach for a variety of datasets and architectures. We hope that our results stimulate further work in this area, including interesting questions regarding hyperparameter optimization and training approaches for multiple TEXP layers, and addressing robustness against strong adversarial attacks in addition to OOD robustness. Furthermore, the recipes of TEXP appear to have similarities with elements of transformer architecture, such as matching queries with keys, and softmax computations. We plan to investigate these connections.

8 Broader impact and limitations

The existing techniques for improving robustness are mostly through the application of data augmentations, changing the optimization loss function, or both. Our approach is complementary, focusing on matching the signal to the input layer of the network, which aligns with the broader goal of making deep networks more interpretable as well. Our experimental results confirm that our method indeed complements standard data augmentation techniques, thereby expanding its applicability to various tasks. A limitation of our work in the current form is that we are yet to develop concrete design guidelines for setting the tilt parameters, which is useful for optimizing the performance of our approach for different architectures. We also note that adding an additional loss function for training results in an increase of the computational cost of training. On the other hand, the computational cost of inference is not impacted substantially, TEXP inference via softmax and thresholding is not significantly more complex than standard ReLU and batch norm. In the future, we will focus on maximizing the effectiveness of our approach across different datasets and models, to ascertain the generalizability. Nonetheless, our findings underscore the value of the tilted exponential layer in enhancing robustness to OOD corruptions, which is important in many practical machine learning tasks where test samples in the real-world are often different from the curated training data.

Acknowledgements

We thank the anonymous reviewers for their valuable suggestions. The work of BP and UM was supported in part by the National Science Foundation under Grants CIF-1909320 and CIF-2224263.

References

Beirami, A., Calderbank, R., Christiansen, M. M., Duffy, K. R., and Médard, M. (2018). A characterization of guesswork on swiftly tilting curves. *IEEE Transactions on Information Theory*.

Butler, R. W. (2007). *Saddlepoint approximations with applications*. Cambridge University Press.

Calian, D. A., Stimberg, F., Wiles, O., Rebuffi, S., György, A., Mann, T. A., and Gowal, S. (2022). Defending against image corruptions through adversarial augmentations. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Carlini, N. and Wagner, D. (2017). Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy*, pages 39–57.

Cekic, M., Bakiskan, C., and Madhow, U. (2022). Neuro-inspired deep neural networks with sparse, strong activations. In *ICIP*, pages 3843–3847. IEEE.

Croce, F., Andriushchenko, M., Sehwag, V., Debenedetti, E., Flammarion, N., Chiang, M., Mittal, P., and Hein, M. (2020). Robustbench: a standardized adversarial robustness benchmark. *arXiv preprint arXiv:2010.09670*.

Croce, F. and Hein, M. (2020). Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International conference on machine learning*, pages 2206–2216. PMLR.

Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., and Le, Q. V. (2019). Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 113–123.

Cubuk, E. D., Zoph, B., Shlens, J., and Le, Q. V. (2020). Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703.

Dembo, A. and Zeitouni, O. (2009). *Large deviations techniques and applications*. Springer Science & Business Media.

Dhillon, G. S., Azizzadenesheli, K., Lipton, Z. C., Bernstein, J., Kossaifi, J., Khanna, A., and Anandkumar, A. (2018). Stochastic activation pruning for robust adversarial defense. *arXiv:1803.01442*.

Dhillon, G. S. and Carlini, N. (2020). Erratum concerning the obfuscated gradients attack on stochastic activation pruning.

Dodge, S. and Karam, L. (2017). A study and comparison of human and deep learning recognition performance under visual distortions. In *2017 26th international conference on computer communication and networks (ICCCN)*, pages 1–7. IEEE.

Gilmer, J., Ford, N., Carlini, N., and Cubuk, E. (2019). Adversarial examples are a natural consequence of test error in noise. In *International Conference on Machine Learning*, pages 2280–2289. PMLR.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2015). Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*.

Hendrycks, D. and Dietterich, T. G. (2018). Benchmarking neural network robustness to common corruptions and surface variations. *arXiv preprint arXiv:1807.01697*.

Hendrycks, D., Mu, N., Cubuk, E. D., Zoph, B., Gilmer, J., and Lakshminarayanan, B. (2020). Augmix: A simple data processing method to improve robustness and uncertainty. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

Huang, T., Halbe, S. A., Sankar, C., Amini, P., Kottur, S., Geramifard, A., Razaviyayn, M., and Beirami, A. (2022). Robustness through data augmentation loss consistency. *Transactions on Machine Learning Research*.

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

Kireev, K., Andriushchenko, M., and Flammarion, N. (2022). On the effectiveness of adversarial training against common corruptions. In *Uncertainty in Artificial Intelligence*, pages 1012–1021. PMLR.

Kort, B. W. and Bertsekas, D. P. (1972). A new penalty function method for constrained minimization. In *IEEE Conference on Decision and Control and 11th Symposium on Adaptive Processes*.

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

Li, T., Beirami, A., Sanjabi, M., and Smith, V. (2021). Tilted empirical risk minimization. In *International Conference on Learning Representations*.

Li, T., Beirami, A., Sanjabi, M., and Smith, V. (2023). On tilted losses in machine learning: Theory and applications. *Journal of Machine Learning Research*, 24(142):1–79.

Liu, G.-H. and Theodorou, E. A. (2019). Deep learning theory review: An optimal control and dynamical systems perspective. *arXiv preprint arXiv:1908.10920*.

Machiraju, H., Choung, O.-H., Herzog, M. H., and Frossard, P. (2022). Empirical advocacy of bio-inspired models for robust image recognition. *arXiv preprint arXiv:2205.09037*.

Madhow, U. (2008). Fundamentals of digital communication.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. (2018). Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations (ICLR)*.

Pee, E. and Royset, J. O. (2011). On solving large-scale finite minimax problems using exponential smoothing. *Journal of Optimization Theory and Applications*.

Qin, Y., Wang, X., Lakshminarayanan, B., Chi, E. H., and Beutel, A. (2023). What are effective labels for augmented data? improving calibration and robustness with autolabel. In *IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*.

Qin, Y., Zhang, C., Chen, T., Lakshminarayanan, B., Beutel, A., and Wang, X. (2022). Understanding and improving robustness of vision transformers through patch-based negative augmentation. *Advances in Neural Information Processing Systems*, 35:16276–16289.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252.

Schneider, S., Rusak, E., Eck, L., Bringmann, O., Brendel, W., and Bethge, M. (2020). Improving robustness against common corruptions by covariate shift adaptation. *Advances in Neural Information Processing Systems*, 33:11539–11551.

Siegmund, D. (1976). Importance sampling in the monte carlo study of sequential tests. *The Annals of Statistics*.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2014). Intriguing properties of neural networks. In *International Conference on Learning Representations (ICLR)*.

Tack, J., Yu, S., Jeong, J., Kim, M., Hwang, S. J., and Shin, J. (2022). Consistency regularization for adversarial robustness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8414–8422.

Yi, M., Hou, L., Sun, J., Shang, L., Jiang, X., Liu, Q., and Ma, Z. (2021). Improved ood generalization via adversarial training and pretraing. In *International Conference on Machine Learning*, pages 11987–11997. PMLR.

Yin, D., Gontijo Lopes, R., Shlens, J., Cubuk, E. D., and Gilmer, J. (2019). A fourier perspective on model robustness in computer vision. *Advances in Neural Information Processing Systems*, 32.

Zagoruyko, S. and Komodakis, N. (2016). Wide residual networks. *arXiv preprint arXiv:1605.07146*.

Zhang, H., Cisse, M., Dauphin, Y., and Lopez-Paz, D. (2018). mixup: Beyond empirical risk management. In *Proc. 6th Int. Conf. Learn. Represent.(ICLR)*, pages 1–13.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model.
   Yes
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm.
   Yes, the training and inference times of our model are compared with baselines.
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries.
   Yes, the source code is available in the repository referenced in Section 5

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results.
   Not Applicable
   (b) Complete proofs of all theoretical results.
   Not Applicable
   (c) Clear explanations of any assumptions.
   Not Applicable

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL).
   Yes, the code is provided, along with instructions to reproduce main results.
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen).
   Yes, please refer to the paragraph on "Our models" in Section 5.
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times).
   Yes, please refer to the paragraph on "Evaluation metrics" in Section 5 for the metrics used in establishing robustness, and Section E for statistical significance of the empirical results.
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider).
   Yes, provided in supplementary materials, Section G.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets.
   Yes, standard open-source image datasets used are cited.
   (b) The license information of the assets, if applicable.
   Not Applicable
   (c) New assets either in the supplemental material or as a URL, if applicable.
   Not Applicable
   (d) Information about consent from data providers/curators.
   Not Applicable
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content.
   Not Applicable.

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots.
   Not Applicable
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable.
   Not Applicable
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation.
   Not Applicable

Appendix

Contents

Page No.

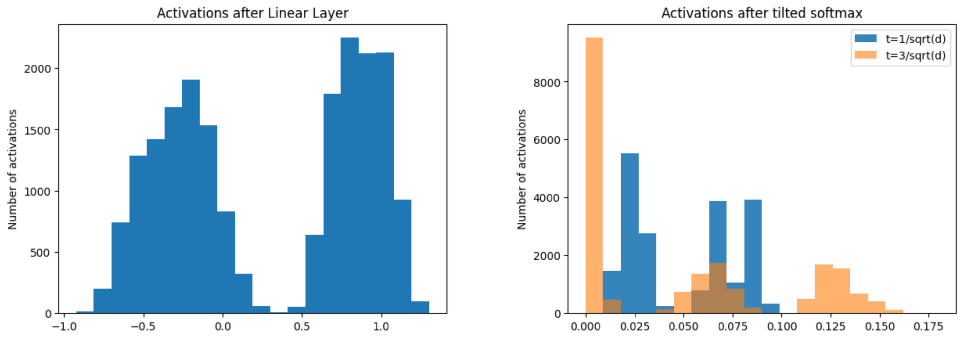
A Geometric insights into TEXP: details on the simplified data models

Recall the data model 1 in Sec. 3, where the input was drawn from

$$p(\mathbf{x}) = \frac{1}{2}\mathcal{N}(\mathbf{x}|\mathbf{s}_1, \sigma^2\mathbf{I}) + \frac{1}{2}\mathcal{N}(\mathbf{x}|\mathbf{s}_2, \sigma^2\mathbf{I})$$

with signals $\mathbf{s}_1 = [1, 0, \ldots, 0]$, $\mathbf{s}_2 = [1/\sqrt{2}, 1/\sqrt{2}, 0, \ldots, 0]$, $d = 10$ and $M = 20$. Under the TEXP objective, depending upon the initializations of the neurons, one or more neurons could align with each of the two signal directions, leading to several "useful" neurons (although not apparent in Fig. 2(a), there are multiple useful neurons in the directions of $\mathbf{s}_1$ and $\mathbf{s}_2$). The activations produced by useful neurons, which are large, survive through the tilted softmax, while the rest are expected to be attenuated.

Focusing on the TEXP objective, we show in Fig. 5 the histograms of the activations (i) at the output of the linear layer (ii) after passing through tilted softmax for two differnet values of $t_{\text{inf}} = 1, 3$, while the training tilt parameter is $t = 10$. We can observe how a stronger tilted softmax can polarize the activations more strongly. In addition, since there could be multiple useful neurons aligned with a signal direction, we could further prune similar neurons, to achieve a more sparse output.

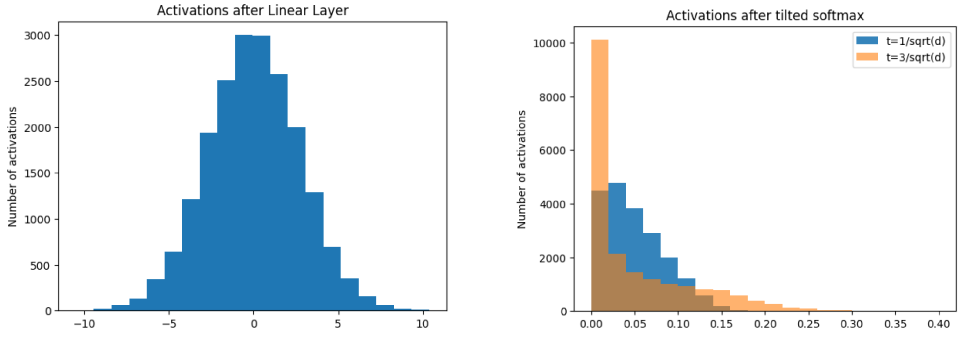


(a) Histogram of activations at the output of linear layer

(b) Histogram of activations after passing through a tilted softmax with $t_{\text{inf}} = 1, 3$.

Figure 5: Simplified data model 1, under TEXP objective

Similarly, for data model 2, the histograms of the activations at the output of the linear layer, and post tilted softmax are shown in Fig. 6. Thus we can observe that TEXP approach encourages sparse activations.



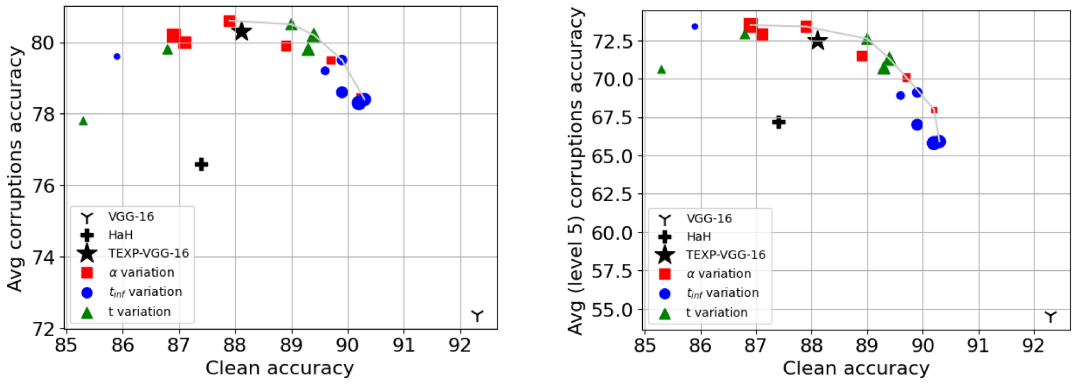
(a) Histogram of activations at the output of linear layer

(b) Histogram of activations after passing through a tilted softmax with $t_{\text{inf}} = 1, 3$.

Figure 6: Simplified data model 2, under TEXP objective

B Sensitivity to TEXP parameters

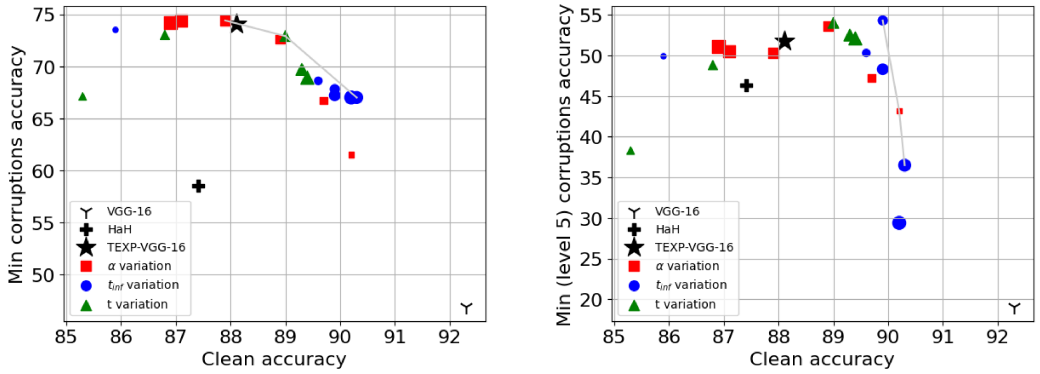
To study the sensitivity of the robustness-accuracy performance to the TEXP parameters, TEXP objective weight α, inference tilt t_{inf} and training tilt t, we perform the following experiment. While keeping two of the three TEXP parameters fixed to those set in the TEXP-VGG-16 model (where we had $\alpha = 0.001$, $t_{\text{inf}} = 1/\sqrt{D} = 0.192$, $t = \beta t_{\text{inf}}$, where $\beta = 10$), we vary the third and train a new TEXP model. We record the clean accuracy and the average corruption robustness (both for all levels and the highest severity level). Fig. 7 shows the robustness-accuracy trade-offs of the various TEXP models, where robustness is measured against corruptions of all severity level (Fig. 7(a)) and highest severity level (Fig. 7(b)). To plot the variation with α, we fix $t_{\text{inf}} = 0.192$, $\beta = 10$, and vary α from 0.00001 to 0.01 (with the larger size of the markers in the figures representing larger values of α, similarly for other variations). For the variation in t_{inf}, we fix $\alpha = 0.001$, $\beta = 10$, and vary t_{inf} from



(a) Average corruption accuracy over all severities

(b) Average (severity level 5) corruption accuracy

Figure 7: Robustness-accuracy trade-offs obtained by varying the TEXP parameters. Increasing marker sizes represent increasing values of the corresponding TEXP parameter. The red markers with increasing sizes encode α variation for $\alpha \in [0.00001, 0.0001, 0.0005, 0.002, 0.005, 0.01]$. The blue markers with increasing sizes encode t_{inf} variation, for $t_{\text{inf}} \in [0.5/\sqrt{D},\ 2/\sqrt{D},\ 3/\sqrt{D},\ 4/\sqrt{D},\ 8/\sqrt{D},\ 16/\sqrt{D}]$. The green markers with increasing sizes encode $t = \beta t_{\text{inf}}$ variation for $\beta \in [1, 5, 15, 25, 50]$.



(a) Minimum corruption accuracy (all severities)

(b) Minimum corruption accuracy for highest severity level

Figure 8: Robustness-accuracy trade-offs obtained by varying the TEXP parameters. Increasing marker sizes represent increasing values of the corresponding TEXP parameter. The red markers with increasing sizes encode α variation for $\alpha \in [0.00001, 0.0001, 0.0005, 0.002, 0.005, 0.01]$. The blue markers with increasing sizes encode t_{inf} variation, for $t_{\text{inf}} \in [0.5/\sqrt{D},\ 2/\sqrt{D},\ 3/\sqrt{D},\ 4/\sqrt{D},\ 8/\sqrt{D},\ 16/\sqrt{D}]$. The green markers with increasing sizes encode $t = \beta t_{\text{inf}}$ variation for $\beta \in [1, 5, 15, 25, 50]$.

$0.5/\sqrt{D}$ to $16/\sqrt{D}$. For the variation in t, we fix $\alpha = 0.001$, $t_{\text{inf}} = 0.192$, and vary β from 1 to 50. We also report the trade-offs with respect to the minimum (worst-case) corruption accuracy in Fig. 8. Overall, our results show that TEXP is mildly sensitive to the choice of the hyperparameters and all variants of TEXP dominate HaH (Cekic et al., 2022) in the Pareto plane.

C TEXP variant with expanded neural competition

In this section, we describe an alternate method of applying the TEXP principles to DNNs. We use the VGG-16 backbone, and change the first layer to a TEXP layer, with a few distinctions in the way tilted softmax, thresholding and TEXP objective is applied.

Similar to the original approach, we implicitly normalize the convolution filter weights to unit ℓ_2 norm as follows:

$$y_i(l) = \frac{(\mathbf{x}(l))^T \mathbf{w}_i}{||\mathbf{w}_i||_2}. \tag{16}$$

Next, instead of promoting competition across the M filters through the tilted softmax, we introduce competition among all the activations in the layer output. Let us index the convolution layer outputs $y_i(l)$, across all filters and spacial locations, by $y_m, m \in [M']$, where $M' = L \times M$. For the running example in the paper with VGG-16 model and CIFAR-10 inputs, we have $M' = 32 \times 32 \times 64$, i.e., the dimension of the layer output post convolution. We then pass these convolution outputs through a tilted softmax, normalizing over the M' scalar layer outputs:

$$p_m = \sigma_m(t_{\text{inf}}\mathbf{y}),$$

where,

$$\sigma_m(\mathbf{x}) = \frac{\exp(x_m)}{\sum_{j=1}^{M'} \exp(x_j)}.$$

and $\mathbf{y} = \{y_m, m = 1, 2, \ldots, M'\}$.

We reindex the post softmax outputs p_m by filter i and spatial location l as $p_i(l)$, and use these notations interchangeably.

Further, in the thresholding block, the thresholds τ_i are set such that for every image, we permit only a certain fraction of the activations, while nullifying the rest. For instance, we set τ_i adaptively such that 20% of the outputs are activated for each image, and each filter. This requires sorting of the activations to decide the thresholds, which makes this approach expensive.

The TEXP objective here is given by

$$\mathcal{L}_{\text{TEXP}} = \frac{1}{t} \log \left(\frac{1}{M'} \sum_{m=1}^{M'} \exp(t a_m) \right) \tag{17}$$

and the *balanced* TEXP objective is

$$\mathcal{L}_{\text{TEXP}}^{\text{bal}} = \frac{1}{t} \log \left(\frac{1}{M'} \sum_{m=1}^{M'} \exp(t(a_m - \bar{a})) \right)$$

where $a_m = \text{ReLU}(y_m)$ are the convolution outputs across all filters and spatial locations in the first layer, passed through a ReLU function, and $\bar{a} = (1/M') \sum_{m=1}^{M'} a_m$ denotes the mean of all the post-ReLU activations in the layer.

We term this variant the TEXP-v2. We set the tilt parameters as $t_{\text{inf}} = 0.1$, $t = 1$ and $\alpha = 0.0001$, while other optimization hyperparameters are retained the same. We observe that augmenting this variant with adversarial training while utilizing the balanced TEXP objective resulted in a model that has a better robustness-accuracy trade-off. We present in Table 4 the test accuracies of this variant contrasted with the matching TEXP model from Section 4. We clarify that all the models listed in the main paper are trained with the (non-balanced) TEXP objective.

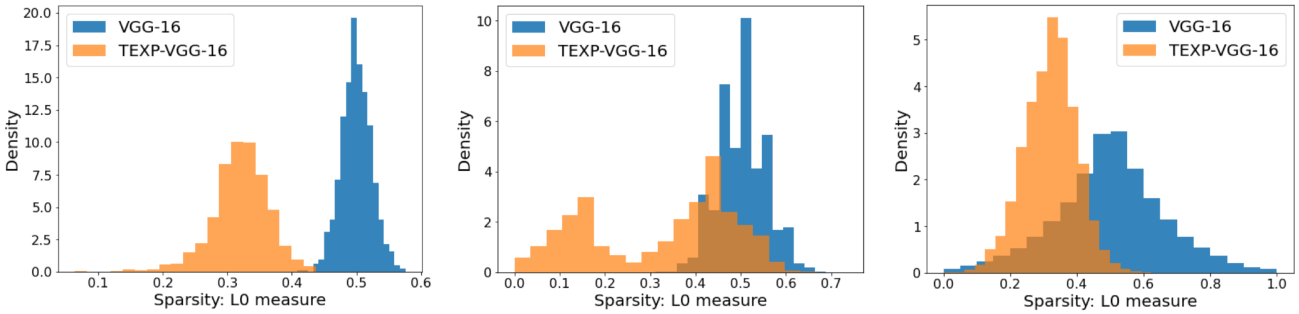
Model	Clean	Noise $\nu = 0.1$	Min/Avg corruptions	Min/Avg severity level: 5	Autoattack ℓ_2 adv, $\epsilon = 0.25$	Autoattack ℓ_∞ adv, $\epsilon = 2/255$
VGG-16	92.3	24.1	46.9/72.4	19.1/54.6	13.2	10.2
TEXP-VGG-16	88.1	75.5	74.1/80.3	51.8/72.5	51.4	42.0
TEXP-v2-VGG-16	88.3	68.4	69.7/79.6	48.3/71.8	39.4	27.6
VGG-16 + Adv Tr	87.8	79.7	51.4/79.1	18.2/70.1	71.5	72.3
TEXP-VGG-16 (Balanced) + Adv Tr	85.8	81.9	67.3/80.3	37.2/74.4	69.4	64.8
TEXP-v2-VGG-16 (Balanced) + Adv Tr	89.0	81.1	78.6/84.1	56.9/79.2	71.8	63.4

Table 4: Robustness to corruptions under the variant TEXP-v2 models on CIFAR-10 clean and corruptions datasets. All numbers reported are test accuracies.

Although this allows us to explicitly control the sparsity levels at the end of first layer and promotes competition among the layer outputs directly, we recommend the TEXP approach described in Section 4 as it aligns more closely to learning matched filters and avoids sorting of activations during training and inference to set the thresholds. For completeness, the average training (inference) time for 1 epoch on the TEXP-v2-VGG-16 model is 31.1 sec (3.19 sec) on an NVIDIA 1080 Ti GPU.

D Sparsity of activations after the first network layer

In this section, we show the statistics of sparsity of the first layer outputs for both VGG-16 and TEXP-VGG-16 models. Recall that in the baseline VGG-16, the ordering is as follows: convolution, followed by ReLU, and then batch-norm. Since batch-norm renders the output completely non-sparse, to draw a fair comparison with TEXP, we obtain sparsity statistics post the ReLU, for VGG-16. For TEXP-VGG-16, we measure the sparsity statistics on the output of the thresholding block in the TEXP layer. We wish to obtain insights on the following questions through experiments: (i) how sparse is the layer output? (ii) for each spatial location, how does neuronal competition give rise to sparsity in the channel/filter dimension? (iii) for each filter, how sparse is the output across the spatial locations? To evaluate these, we take a batch of 1000 CIFAR-10 test images, and compute an L_0 notion of sparsity on the first layer output as the number of non-zero entries (measured by counting activations larger than some small epsilon) normalized by the number of scalar dimensions. For instance, to answer the first question, given an image, we count the number of non-zeros over the entire layer output and normalize by $C \times H \times W$. For the second question, for every spatial location, we count the non-zeros along channel dimension, and normalize by the number of channels C. For the third, for every filter, we count the non-zeros in spatial dimensions and normalize by $H \times W$. We then plot the histograms of the sparsity levels obtained for the three cases in Fig. 9. We observe that TEXP leads to sparse layer outputs, as expected, due to the neuronal (and spatial) competition from TEXP inference and training.



(a) Sparsity in $C \times H \times W$ dimensions (b) Sparsity in channel/filter dimension (c) Sparsity in spatial dimensions

Figure 9: Histograms of the sparsity levels in the first layer output (lower values indicate more sparse).

E Statistical significance of the results

The results in Table 1 on the performance of all the VGG based models on CIFAR-10 dataset are reported after averaging across five runs. The numbers reported are the sample mean values of the accuracies across the five runs ± standard error on the mean. The standard error is given by $\sigma/\sqrt{N}$, where σ is the sample standard deviation and N is the number of runs.

F Ablation study

In this section, we contrast the contributions of the TEXP objective and components of the inference paths in comparison to the baseline model. We show in Table 5 four VGG-16 based models: (i) Baseline VGG; (ii) VGG with the adpative thresholding block replacing the ReLU in the first layer (named VGG-16 + Thresholding), which is similar to a TEXP inference layer albeit with retaining the batchnorm; (ii) TEXP-VGG-16-Inference-only, which introduces the tilted softmax in place of the batchnorm (and with $\alpha = 0$, i.e. no TEXP objective); (iii) typical TEXP-VGG-16, which introduces the TEXP objective with $\alpha > 0$ over the inference-only structure. We can observe that addition of each of the components of the TEXP approach improves the robustness of the model.

Although there have been works in literature that approach robustness through stochastic pruning techniques, we observe that activation pruning alone through the thresholding block (which performs deterministic activation pruning), has a limited impact on robustness. However, when the thresholding and tilted softmax blocks are combined with the TEXP objective, our approach aligns filters to match the incoming patterns and shapes the activations, ensuring that there is some fraction of strong activations that survive the thresholding.

Model	Clean	Noise $\nu = 0.1$	Min/Avg corruptions	Min/Avg severity level: 5	Autoattack ℓ_1 adv, $\epsilon = 3$	Autoattack ℓ_2 adv, $\epsilon = 0.25$	Autoattack ℓ_∞ adv, $\epsilon = 2/255$
VGG-16	92.3	24.1	46.9/72.4	19.1/54.6	10.1	13.2	10.2
VGG-16 + Thresholding	91.6	39.1	53.1/73.8	18.4/55.7	15.1	20.5	16.1
TEXP-VGG-16-Inference-only	90.0	60.8	63.1/78.7	44.0/68.3	37.0	36.3	24.5
TEXP-VGG-16	88.1	75.5	74.1/80.3	51.8/72.5	47.2	51.4	42.0

Table 5: Ablation study of TEXP components on CIFAR-10 clean and corruption datasets. All numbers reported are test accuracies for a single run.

G Computation time

All the experiments on the CIFAR-10 dataset were mainly performed using 4 NVIDIA GeForce 1080 Ti GPUs with 12GB memory. For experiments on the CIFAR-100 and ImageNet-1K datasets, we relied on an NVIDIA A100 GPU with 80GB memory. The training and inference times for the baseline VGG-16, HaH and TEXP-VGG-16 models are contrasted in Table 6. To train HaH models, we use the repository by Cekic et al. (2022), and utilize the set of hyperparameters outlined in the paper.

GPU →	1080Ti		A100	
Models ↓	Training	Inference	Training	Inference
VGG-16	18.9±0.27	2.79±0.02	5.9±0.08	1.13±0.006
HaH (Cekic et al., 2022)	107.3±0.38	4.00±0.04	34.6±0.06	1.41±0.006
TEXP-VGG-16	27.5±0.18	2.84±0.01	8.23±0.15	1.12±0.009

Table 6: Average training (for 1 epoch) and inference times in seconds. The numbers reported are average time ± standard error in mean, where standard error = sample standard deviation/$\sqrt{N}$.