A. Encoding Network Modules

This section provides the implementation details for the modules constituting our specification encoding network as discussed in §3.1. A typical Fourier domain augmentation pipeline can look as follows:

$$x \to \mathcal{F} \to \text{FFTshift} \to +\hat{\delta}(\text{perts}) \to \text{IFFTshift} \to \bar{\mathcal{F}}^1 \to x'.$$

The FFTshift operation shifts the zero-frequency component to the center of the DFT spectrum, thereby offering a more interpretable view of the spectrum for Fourier analysis and perturbation set design, with the IFFTshift operation reverting this spectrum shift afterwards. To avoid the IFFTshift operation in the verification path, the $\hat{\delta}$ defined as part of the specification design is manually IFFT-shifted and then fed to the encoding network. Therefore, the encoding modules explained in this section expect their DFT coefficient space inputs to have been IFFT-shifted and not centered around the lowest frequency. Also, the DFT coefficients are fed to the encoding network as a stacked vector of its Cartesian coordinates $\begin{bmatrix}\Re\hat{x}\\ \Im\hat{x}\end{bmatrix}$. Overall the encoding pipeline for single input specifications is:

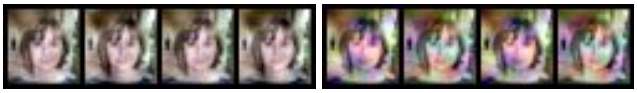
$$E(\hat{\zeta}, x) = \text{CN} \circ \text{PM}(x) \circ \text{IDFT} \circ \text{CBCT}(\hat{\zeta}),$$

and for the two input-conditional specifications is:

$$E(\hat{\zeta}_1, \hat{\zeta}_2') = \text{CN} \circ \text{IDFT} \circ \text{SI}(\hat{\zeta}_1, \hat{\zeta}_2').$$

The modules featuring in E above are detailed below.

Channel Broadcaster (CB) module The purpose of this module is to encode white-light illumination changes. It is constructed using a broadcasting, unit-valued, 1×1 Convolution layer, i.e., a convolution layer with kernel size and kernel weight of one, no bias, one input filter that accepts the perturbation to broadcast and the output filters equal to the number of color channels in the input data. The effect of including this module in the encoding network can be visually verified from Figure 7.



Perturbed images with Channel Broadcasting Perturbed images without Channel Broadcasting

Figure 7. Perturbed images with and without Channel Broadcasting. It could be used when generalizing against whitelight illumination changes, and may be skipped when generalizing against color-tone changes that are more common in domain changes.

Conjugate Transpose (CT) module This module can be optionally included in the encoding network when dealing with real-valued perturbations. Given the property that the DFT of real-valued is conjugate symmetric, this module constructs the complete Fourier domain perturbation set from one half of the Fourier coefficients specified to define an input perturbation set. For this, it uses flip operations to mirror the specified coefficients in Cartesian form $\hat{\delta} \in \mathbb{R}^{N\times N} = \{\Re(\delta), \Im(\delta)\}$ and N is odd, as follows,

$$\text{CT}(\hat{\delta}_{[:\frac{N}{2},:\frac{N}{2}]}) = \left[\begin{array}{c|c|c} \hat{\delta}_{[0,0]} & \hat{\delta}_{[0,1:\frac{N}{2}]} & f^c_x(\hat{\delta}_{[0,1:\frac{N}{2}]}) \\ \hline \hat{\delta}_{[1:\frac{N}{2},0]} & \hat{\delta}_{[1:\frac{N}{2},1:\frac{N}{2}]} & f^c_{xy}(\hat{\delta}_{[1:\frac{N}{2},1:\frac{N}{2}]}) \\ \hline f^c_y(\hat{\delta}_{[1:\frac{N}{2},0]}) & & \end{array}\right],$$

where $f^c_{\text{axes}}(x)$ denotes a flip operation on x along the axes specified in subscript along with multiplication by the constant $c = \{1, -1\}$ for the real and imaginary components of $\hat{\delta}$ resp. Since we already ensured low-dimensionality for our Fourier domain specifications by the use of the mask $\hat{M}$ (2), we did not use this module for further dimensionality reduction in our verification experiments.

The CB and CT modules described above are together referred to as CBCT in our encoding network definition.

IDFT module The purpose of the IDFT module is to implement the Inverse Discrete Fourier Transform as a network module that can be prepended to a network to be verified and allow bound propagation through it by the SoA verifiers. It is the main module of the proposed encoding network and features in the verification path for all specifications. The input to the module is a nonuniform interval set in the DFT coefficient space, expressed as the stacked vector of its Cartesian planes $\hat{\zeta} := \begin{bmatrix}\Re(\hat{\delta})\\ \Im(\hat{\delta})\end{bmatrix}$. Since most verifiers do not support bound propagation through Complex-valued layers, we implement the IDFT using two real-valued linear layers. Formally, we implement $2d$-IDFT as:

$$\delta \in \mathbb{R}^{N\times N} = \Re\left(\Omega^1_N \hat{\zeta} \Omega^{1T}_N\right), \tag{6}$$

where following (I)DFT definitions in §2,

$$\Omega^c_N = \frac{1}{C}\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega_c & \omega_c^2 & \cdots & \omega_c^{N-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega_c^{N-1} & \omega_c^{2\times(N-1)} & \cdots & \omega_c^{(N-1)\times(N-1)} \end{bmatrix},$$

where $\omega_c^{k\times l} = e^{\iota c 2\pi(\frac{k}{H}+\frac{l}{W})}$ with $c = -1, 1$ and $C = N^2, 1$ for DFT and IDFT respectively. The inner $\Omega^1_N\hat{\zeta}$ is implemented using the Complex multiplication property, $\hat{x}_1\hat{x}_2 = (\Re(\hat{x}_1)\Re(\hat{x}_2) - \Im(\hat{x}_1)\Im(\hat{x}_2)) + \iota(\Im(\hat{x}_1)\Re(\hat{x}_2) + \Re(\hat{x}_1)\Im(\hat{x}_2))$ [5], as:

$$\hat{\zeta}_{\text{intm.}} = \begin{bmatrix}\Re(\hat{\delta}_{\text{intm.}})\\ \Im(\hat{\delta}_{\text{intm.}})\end{bmatrix} \in \mathbb{R}^{2N\times N} := \Omega^1_N\hat{\zeta} = \begin{bmatrix}\Re(\Omega^1_N) & -\Im(\Omega^1_N)\\ \Im(\Omega^1_N) & \Re(\Omega^1_N)\end{bmatrix}\begin{bmatrix}\Re(\hat{\delta})\\ \Im(\hat{\delta})\end{bmatrix},$$

and thereafter, $\Re(\hat{\delta}_{\text{intm.}}\Omega^{1T}_N)$ is implemented as the following dot product:

$$\delta = \Re(\hat{\delta}_{\text{intm.}})\Re(\Omega^1_N) - \Im(\hat{\delta}_{\text{intm.}})\Im(\Omega^1_N).$$

In the shared codebase, we validate the above implementation of IDFT by testing against the IDFT function *numpy.fft.ifft2* provided by the popular Numpy library [14].

Clipping module Clipping of an image x to the $[0,1]$ range is implemented as,

$$C(x) = 1 - \lfloor 1 - \lfloor x \rfloor \rfloor = \begin{cases} 1, & x \geq 1, \\ x, & 0 < x < 1, \\ 0, & x \leq 0, \end{cases}$$

where $\lfloor x \rfloor$ is equivalent to and implemented using the ReLU function, which is supported by most NN verifiers.

Segment Interpolator (SI) module This module takes in two inputs $i_1 := |\mathcal{F}(x_1)|e^{\angle\mathcal{F}(x_1)}$ and $i_2 := |\mathcal{F}(x_2)|e^{\angle\mathcal{F}(x_1)}$ where x_1 is the image into which the style elements from image x_2 should be injected. However, this injection should not overly modify the structural content of the interpolated images from that of image x_1. To ensure the latter, notice that the phase of both the inputs to this module is the phase of x_1 image, i.e., $e^{\angle\mathcal{F}(x_1)}$. An interpolation set between these inputs is constructed using a linear layer with weight as $(i_2 - i_1)^T$ and bias as i_1. The input to this layer is a scalar $\alpha \in [0,1]$ and results in outputs of the form $x' = \alpha(i_2 - i_1) + i_1$. Therefore, increasing α increases the visual or perceptible closeness of the interpolated images from image i_1 to image i_2.

Domain Resolution Setting The purpose of this optional setting is to construct pixel-space perturbations having lower frequencies than the fundamental frequency of the input image, while requiring few DFT coefficients. For this, we increase the resolution of the Fourier domain, or equivalently the scale of the pixel domain, by a factor r. The IDFT matrix for this higher frequency-resolution resolution Ω_{rN} is as follows:

$$\Omega_{rN} = \begin{bmatrix} \Omega^{rN}_{0,0} & \Omega^{rN}_{0,1} & \cdots & \Omega^{rN}_{0,r-1} \\ \Omega^{rN}_{1,0} & \Omega^{rN}_{1,1} & \cdots & \Omega^{rN}_{1,r-1} \\ \vdots & \vdots & \ddots & \vdots \\ \Omega^{rN}_{r-1,0} & \Omega^{rN}_{r-1,1} & \cdots & \Omega^{rN}_{r-1,r-1}, \end{bmatrix}$$

where $\Omega^{rN}_{i,j} = \Omega_{rN[i\times N:(i+1)\times N,\ j\times N:(j+1)\times N]}$. Therefore, from (6), the higher scale pixel-space perturbation should be $\delta_{rN} = \Re(\Omega_{rN}\hat{\delta}_{rN}\Omega^T_{rN})$, where $\delta_{rN} \in \mathbb{R}^{rN\times rN}$ and $\hat{\delta}_{rN} \in \mathbb{C}^{rN\times rN}$. While one could construct the higher scale, i.e., $rN \times rN$-sized, perturbation set using (6), and use its $N \times N$ cropped set as the final perturbation set, we additionally simplify the above relation. Given the fact that our DFT space perturbations are zero-centered, and the purpose of increasing frequency domain resolution is to efficiently capture perturbations of frequencies less than ω_f, we assume that at most $d < N$ frequency coefficients are perturbed. This implied that $\hat{\delta}_{rN[i\times N:(i+1)\times N,\ j\times N:(j+1)\times N]} = 0$ for $i,j \notin \{0, r-1\}$, and

$$\begin{aligned} \delta_{rN} &= \Omega_{rN}\hat{\delta}_{rN}\Omega^T_{rN} \\ &= \Omega_{rN}\begin{bmatrix} \hat{\delta}_{0,0} & 0 & \cdots & \hat{\delta}_{0,r-1} \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \hat{\delta}_{r-1,0} & 0 & \cdots & \hat{\delta}_{r-1,r-1} \end{bmatrix}\Omega^T_{rN}. \end{aligned}$$

Since, we only require $\delta_{0,0} \in \mathbb{R}^{N\times N}$, it can be obtained by:

$$\begin{aligned} \delta = \delta_{0,0} = \Omega^{rN}_{0,0}\hat{\delta}_{0,0}\Omega^{TrN}_{0,0} \quad &\text{(similar to } \delta \text{ in (6))} \\ + (\Omega^{rN}_{0,r-1}\hat{\delta}_{r-1,0}\Omega^{TrN}_{0,0} &+ \Omega^{rN}_{0,0}\hat{\delta}_{0,r-1}\Omega^{TrN}_{r-1,0} \\ + \Omega^{rN}_{0,r-1}\hat{\delta}_{r-1,r-1}&\Omega^{TrN}_{r-1,0}). \end{aligned}$$

Thus, when frequency domain resolution is increased to encode smooth perturbations of frequency less than ω_f and $d < N$ coefficients are perturbed, the IDFT output is the usual IDFT output term summed with three additional terms. Each of these terms can be computed using the same implementation as used for $\Omega^{rN}_{0,0}\hat{\delta}_{0,0}\Omega^{TrN}_{0,0} \equiv \Omega^1_N\hat{\delta}\Omega^{1T}_N$ in (6).

B. Fourier Domain Specification Design

The typical norm-bounded local set in spatial domain capturing additive perturbations can be written as:

$$\begin{aligned} \mathcal{B}_{p,\epsilon}(x) &= \{x' \mid ||x' - x||_p \leq \epsilon\}, \\ &= \{x + \delta \mid ||\delta||_p \leq \epsilon\}, \\ &= x + \{\delta \mid ||\delta||_p \leq \epsilon\} = x + \mathcal{B}_{p,\epsilon}(\mathbf{0}), \end{aligned}$$

For additive perturbation model, we define the frequency domain augmentation set as

$$\mathcal{B}_{p,\hat{\epsilon}}(x) = \{x + \bar{\mathcal{F}}^1(\hat{\delta}) \mid \hat{\delta} \in \mathcal{B}_{p,\hat{\epsilon}}(0)\}.$$

We now provide the theorem proofs omitted from the main paper.

Theorem 1 *[$\mathcal{B}_{\infty,\hat{\epsilon},\hat{M}} \to \mathcal{B}_{\infty,\epsilon}$] The smallest pixel space ℓ_∞-ball $\mathcal{B}_{\infty,\epsilon}(0)$ enclosing the output set of IDFT for input set $\mathcal{B}_{\infty,\hat{\epsilon},\hat{M}}(0)$ has $\epsilon_\infty = d\hat{\epsilon}_\infty$, where d is the maximum number of DFT coefficients allowed to perturb by mask $\hat{M}$, i.e., $\max_{\hat{x}\in\mathcal{B}_{\infty,\frac{\epsilon_\infty}{d},\hat{M}}(0)} ||\bar{\mathcal{F}}^1(\hat{x})||_\infty = \epsilon_\infty$.*

Proof of Theorem 1 Let us denote the i-th row of the IDFT matrix Ω^{-1}_N by $o^{-1,i}_N$. Using the definition of ℓ_∞ norm, we can write:

$$\max_{\hat{x}\in\mathcal{B}_{\infty,\frac{\epsilon_\infty}{d},\hat{M}}(0)}$$

All the entries of Ω^{-1}_N have unit magnitude. Therefore, for any $i \in \{0,\ldots,D-1\}$, we can use the triangle inequality to obtain:

$$\begin{aligned} |o^{-1,i}_N\hat{x}| = |\sum_{j=0}^{D-1} o^{-1,i}_N[j]\hat{x}[j]| \leq \sum_{j=0}^{D-1} |o^{-1,i}_N[j]\hat{x}[j]| = \\ = \sum_{j=0}^{D-1} |o^{-1,i}_N[j]||\hat{x}[j]| = \sum_{j=0}^{D-1} |\hat{x}[j]|. \end{aligned} \tag{7}$$

Hence:

$$\max_{\hat{x}\in\mathcal{B}_{\infty,\frac{\epsilon_\infty}{d},\hat{M}}(0)} \|\bar{\mathcal{F}}^1(\hat{x})\|_\infty \leq \max_{\hat{x}\in\mathcal{B}_{\infty,\frac{\epsilon_\infty}{d},\hat{M}}(0)} \sum_{j=0}^{D-1} |\hat{x}[j]| = \\ = \sum_{j=0}^{D-1} \hat{M}[j]\frac{\epsilon_\infty}{d} = \epsilon_\infty.$$

In order to get a lower bound on $\max_{\hat{x}\in\mathcal{B}_{\infty,\frac{\epsilon_\infty}{d},\hat{M}}(0)} \|\bar{\mathcal{F}}^1(\hat{x})\|_\infty$, let us choose $\hat{x}' = \left(\frac{\epsilon_\infty}{d}\hat{M}\right) \in \mathcal{B}_{\infty,\frac{\epsilon_\infty}{d},\hat{M}}(0)$. For $i = 0$, $|o_N^{-1,i}\hat{x}'| = |\sum_{j=0}^{D-1} \frac{\epsilon_\infty}{d}\hat{M}[j]| = \epsilon_\infty$. Equation (7) applies for $i \in \{1,\ldots,D-1\}$, yielding $|o_N^{-1,i}\hat{x}'| \leq \sum_{j=0}^{D-1} \hat{M}[j]\frac{\epsilon_\infty}{d} = \epsilon_\infty$. Thus, $\|\bar{\mathcal{F}}^1(\hat{x}')\|_\infty = \epsilon_\infty$. Therefore, we have

$$\epsilon_\infty = \|\bar{\mathcal{F}}^1(\hat{x}')\|_\infty \leq \max_{\hat{x}\in\mathcal{B}_{\infty,\frac{\epsilon_\infty}{d},\hat{M}}(0)} \|\bar{\mathcal{F}}^1(\hat{x})\|_\infty \leq \epsilon_\infty,$$

concluding the proof.

Theorem 2 *[$\mathcal{B}_{2,\hat{\epsilon},\hat{M}} \rightarrow \mathcal{B}_{2,\epsilon}$] The output of IDFT for input $\mathcal{B}_{2,\hat{\epsilon},\hat{M}}(0)$ is the pixel space ℓ_2-ball $\mathcal{B}_{2,\epsilon}(0) \in \mathbb{R}^D$ with $\epsilon_2 = \sqrt{D}\hat{\epsilon}_2$, i.e., $\{\bar{\mathcal{F}}^1(x) \mid x \in \mathcal{B}_{2,\frac{\epsilon_2}{\sqrt{D}},\hat{M}}(0)\} = \mathcal{B}_{2,\epsilon_2}(0)$.*

Proof of Theorem 2 The result follows from the Parseval's theorem on the equivalence of intensities in the pixel and Fourier domain up to a constant factor, and for the DFT-IDFT definition pair in equation (1) can be derived as follows. Let $\hat{\delta} \in \mathcal{B}_{2,\hat{\epsilon}}(0)$ and $\delta = \bar{\mathcal{F}}^1(\hat{\delta})$, then we have:

$$\begin{aligned}
\|\hat{\delta}\|_2^2 &= \sum_{k=0}^{H-1}\sum_{l=0}^{W-1} |\hat{\delta}[k,l]|^2 = \sum_{k=0}^{H-1}\sum_{l=0}^{W-1} \hat{\delta}[k,l]\hat{\delta}^*[k,l] \\
&= \sum_{k=0}^{H-1}\sum_{l=0}^{W-1} \hat{\delta}[k,l]\left(\frac{1}{WH}\sum_{n=0}^{H-1}\sum_{m=0}^{W-1} \delta[m,n]e^{-i2\pi(\frac{kn}{H}+\frac{lm}{W})}\right)^* \\
&= \frac{1}{WH}\sum_{n=0}^{H-1}\sum_{m=0}^{W-1} \delta^*[m,n]\left(\sum_{k=0}^{H-1}\sum_{l=0}^{W-1} \hat{\delta}[k,l]e^{i2\pi(\frac{kn}{H}+\frac{lm}{W})}\right) \\
&= \frac{1}{WH}\sum_{n=0}^{H-1}\sum_{m=0}^{W-1} \delta^*[m,n]\delta[m,n] \\
&= \frac{1}{WH}\sum_{n=0}^{H-1}\sum_{m=0}^{W-1} |\delta[n,m]|^2 = \frac{1}{WH}\|\delta\|_2^2. \qquad (8)
\end{aligned}$$

Thus, for a pixel space ℓ_2-norm perturbation epsilon ϵ, we have:

$$\begin{aligned}
\mathcal{B}_{2,\epsilon}(0) &= \{\delta \mid \|\delta\|_2 \leq \epsilon\}, \\
&= \left\{\delta \mid \|\hat{\delta}\|_2 \leq \frac{1}{\sqrt{WH}}\epsilon\right\}, \quad \text{from (8)} \\
&= \mathcal{B}_{\hat{\epsilon},2}(0) \implies \hat{\epsilon}_2 = \frac{1}{\sqrt{WH}}\epsilon_2.
\end{aligned}$$

Therefore, we have $\hat{\epsilon}_2 = \frac{1}{\sqrt{WH}}\epsilon_2$, concluding the proof.

Sound translation of Fourier specifications from Polar to Cartesian form. Given the limits on the amplitude $[|\hat{\delta}|_l, |\hat{\delta}|_u]$ and phase $[\angle\hat{\delta}_l, \angle\hat{\delta}_u]$ of a Fourier coefficient $\hat{\delta}$, and the corresponding four Cartesian coordinates $\{z_{ll} \hat{=} |\hat{\delta}|_l\angle\hat{\delta}_l, z_{lu} \hat{=} |\hat{\delta}|_l\angle\hat{\delta}_u, z_{ul} \hat{=} |\hat{\delta}|_u\angle\hat{\delta}_l, z_{uu} \hat{=} |\hat{\delta}|_u\angle\hat{\delta}_u\}$, one has:

$$\Re(\hat{\delta})_l = \begin{cases} -|\hat{\delta}|_u, & \text{if } \pi \in [\angle\hat{\delta}_l, \angle\hat{\delta}_u], \\ \min\{\Re(z_{ll}), \Re(z_{lu}), \Re(z_{ul}), \Re(z_{uu})\} \end{cases}$$

$$\Re(\hat{\delta})_u = \begin{cases} |\hat{\delta}|_u, & \text{if } 0 \in [\angle\hat{\delta}_l, \angle\hat{\delta}_u], \\ \max\{\Re(z_{ll}), \Re(z_{lu}), \Re(z_{ul}), \Re(z_{uu})\} \end{cases}$$

$$\Im(\hat{\delta})_l = \begin{cases} -|\hat{\delta}|_u, & \text{if } \frac{3\pi}{2} \in [\angle\hat{\delta}_l, \angle\hat{\delta}_u], \\ \min\{\Im(z_{ll}), \Im(z_{lu}), \Im(z_{ul}), \Im(z_{uu})\} \end{cases}$$

$$\Im(\hat{\delta})_u = \begin{cases} |\hat{\delta}|_u, & \text{if } \frac{\pi}{2} \in [\angle\hat{\delta}_l, \angle\hat{\delta}_u], \\ \max\{\Im(z_{ll}), \Im(z_{lu}), \Im(z_{ul}), \Im(z_{uu})\} \end{cases}$$

The above Cartesian space bounds are sound, i.e., they enclose all frequency coefficient values that can be reached by the given amplitude and phase set. However, they are sub-optimal as they allow all combinations of the real and imaginary components enclosed within, and therefore introduce over-approximation in the otherwise precise encoding.

C. Experiment Details

Visual demonstrations and implementation for our encodings, unit tests, attacks, verification and certified training for the proposed specifications is available at https://github.com/hh10/Fourier-Verification-and-Certified-Training.

Network details. The network used in this work is the CNN7 convolutional network, owing to its typical use in most existing verification and certified training works, starting with [30]. For our 10 class datasets, this network has 5×{Conv(3x3) + Batch Normalization + ReLU} + Flatten + Linear(512) + ReLU + Linear(10) layers amounting to 62k ReLUs.

Platform details. The network trainings and verification were run on an Nvidia GeForce RTX 3090 GPU with 25 GB VRAM to allow a consistent comparison among the bound computation and the training times reported in Figure 6.

Specification and Verification details. For verification of single input specifications, we choose to report for additive specifications as per 2 as most existing verification works verify for additive noise. We picked the typically reported pixel-domain ϵ_∞ values of 1/255, 2/255 and 8/255 for our specifications. To define the Fourier domain specifications of varying dimensionality in a consistent manner, we computed the $\hat{\epsilon}_\infty$s corresponding to ϵ_∞s as per Theorem 1. After designing all elements of the input specification $\mathcal{B}_{p,\hat{\epsilon},\hat{M}}$ as discussed in §3.2, the verification of the augmented network $N_\theta \circ E(x)$ for this specification was done using AutoLirpa.

Training details. The certified trainings were done for the same specifications that the non-robustly trained networks were verified for. All trainings were specified using a configuration file that specifies parameters such as the experiment seed, dataset transforms, training epochs, optimizer parameters such as learning rate and weight decay, perturbation growing schedule, etc. The default values of these parameter for most trainings were: i) training epochs were 150-175, ii) the optimizer settings were kept consistent with that in [30], including the use of Adam optimizer with learning rate of $5e-4$, iii) the perturbation epsilon growth profile was S-shaped ($\epsilon(\text{epoch}) = \epsilon_l + (\epsilon_u - \epsilon_l)\frac{e^\beta}{e^\beta + (1-e)^\beta}$, $e = \frac{\text{epoch}}{\#\text{epochs}}$, $\beta = 1.5$), starting from epoch 11 till 140. The reported trained models, the configuration files used for their training and evaluation, and the instructions to reproduce the reported experiments are provided in the shared codebase.

Certified Training. The certified trainings for our Fourier domain specifications for networks evaluated in Fig 6c-Fig 6d involved

$$\min_\theta \mathbb{E}_{(x,y)\in(\mathcal{X},\mathcal{Y})}\left[\max_{\hat{\delta},\alpha} L\left(N_\theta \circ E(x)(\hat{\delta},\alpha), y\right)\right],$$

where the inner worst case loss is computed using bounds on the network output. The training epochs were 150 for FB trainings and ≈ 500 for our Fourier domain IBP trainings. The IBP trainings were trained for more epochs to grow the perturbation epsilon more gradually than the other trainings that induce relatively less over-regularization. For faster convergence of Fourier domain IBP trainings, we use the specialized weight initialization and regularization from [30]. The pixel domain certified network was trained as suggested in [24] and using their codebase.

Adversarial Training and Attacks. The pixel-domain adversarial training for the network evaluated in Fig 5b involved

$$\min_\theta \mathbb{E}_{(x,y)\in(\mathcal{X},\mathcal{Y})}\left[L\left(N_\theta(x_{adv}), y\right)\right],$$

where $x_{adv} \in \mathcal{B}_{\infty,\epsilon}$ is the pixel-domain adversarial attack such that $N_\theta(x_{adv}) \neq y$. Similarly, the Fourier-domain adversarial attack for the single input specifications is $\hat{\delta}_{adv} \in \mathcal{B}_{\infty,\hat{\epsilon},\hat{M}}$ and for the two input conditional specifications is $\alpha_{adv} \in [0,1]$ such that the network N_θ's output for the image generated by $E(x)$ for $\hat{\delta}_{adv}$and α_{adv} is incorrect, i.e., $N_\theta \circ E(x)(\hat{\delta}_{adv}, \alpha_{adv}) \neq y$. We considered the popular PGD-based attack, implemented as in the recent work [48], for adversarial training and to evaluate the adversarial accuracy of the networks. *Projected Gradient Descent (PGD)* [21]-based attacks, such as used in [2], take steps in the direction of the sign of the gradient of the loss w.r.t. to the specification variables $\nabla_{\hat{\delta}'} L(N_\theta \circ E(\hat{\delta}'), y)$ while projecting onto the input specification $\mathcal{B}_{\infty,\hat{\epsilon},\hat{M}}$ after every iteration. The steps are taken for a fixed number of iterations or until reaching a $\hat{\delta}'$ that results in network incorrectness. The default parameters used to find this attack in $\mathcal{B}_{\infty,\hat{\epsilon},\hat{M}}$ were: number of iterations=20, step size (α)=$\frac{\epsilon}{5}$. Our codebase also provide the implementation of the Low-Frequency Backdoor Attack (LFBA) used in [25], though we found it to be a weaker attack than PGD for our specifications, therefore we only report the PGD-based adversarial accuracy in Fig. 6. With both the attacks, the verified accuracy using both complete and incomplete verifiers lower bounded the adversarial accuracy.

Augmented Training. Augmented trainings for our Fourier domain specifications involved

$$\min_\theta \mathbb{E}_{(x,y)\in(\mathcal{X},\mathcal{Y})}\left[L\left(N_\theta \circ E(x)(\hat{\delta}', \alpha'), y\right)\right],$$

where $\hat{\delta}' \in \mathcal{B}_{p,\hat{\epsilon},\hat{M}}$ for the single input specifications and $\alpha' \in [0,1]$ for the two input conditional specifications. The standard augmentation training to prepare the network evaluated in Fig 5a is trained with the common augmentations provided by PyTorch transforms, i.e. random horizontal flip (probability=0.5), color jitter (brightness=(0.9, 1.15), contrast=(0.8, 1.2), saturation=(0.9, 1.1), hue=(-0.05, 0.05)) and random grayscale (probability=0.2).

D. Additional Experiments

D.1. Robustness verification of conditional specifications.

Along with verifying single input-specifications in §4.2, we verify two inputs-based conditional specifications similar to the ones shown in Fig. 4 (c). We train two CNN7 networks (A and B) on CODaN dataset; A is trained with the original day images from the dataset, B is trained with day images that are augmented in the frequency domain to appear like night images as per [43]. Both networks are then empirically tested and verified for object classification in image sets that span from the Fourier amplitudes of day images towards those of unseen night images. Results for the same are reported in Figure 8 and show the verified accuracy lower bounding the adversarial accuracy for both networks. While the verified accuracy for the network trained with frequency domain augmentation is higher than for the normally trained network as expected, the perturbation magnitudes that could be verified as robust are small for both. This can be explained given that a day to night scene change incurs a large intensity change and these models are not certifiably trained for conditional specifications.

Certifiable training for conditional specification offers implementation challenge as the prepended encoding network needs to be reconstructed for every input batch. This reconstruction incurs significant time which is acceptable

for verification but not for training. Therefore, we leave the certified training for conditional specifications for future work.

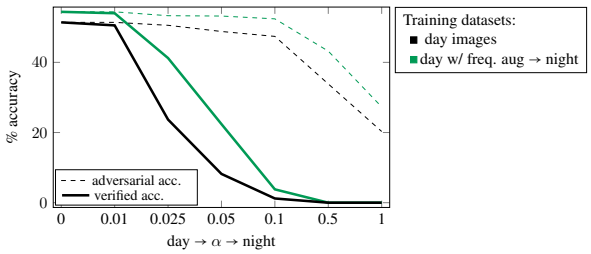


Figure 8. Verification outcomes for conditional domain change specification $X_{\text{des,cond}}(\alpha, X, X')$ in (5) for the 10-class CODaN dataset, with X, X' being the day and night image datasets resp.

D.2. Extending to other Frequency domain transforms

This work primarily focused on DFT as the frequency transform to decompose data in terms of frequencies. However, the proposed framework can readily support any transform that can be computed through matrix multiplications. Therefore, next we discuss the straightforward extension of the proposed approach for verifying specifications defined in the coefficient space of another popular frequency transform, i.e., the Discrete Cosine Transform (DCT).

DCT. Similar to DFT, the DCT of an input $x \in \mathbb{R}^{W\times H}$ can be performed using the DCT matrix $F \in \mathbb{R}^{W\times H}$ as $\hat{x} \in \mathbb{R}^{W\times H} := FxF^T$, with its Inverse DCT being $x = F^T\hat{x}F$ if F is orthogonal. Thus, IDCT can be implemented using linear layers with F matrix as its weights and no bias. There exists different definitions of DCT transforms, matrices and normalizations; we implement the most commonly used Type-II IDCT with orthonormalization as our encoding network E. We check the correctness of E's implementation of IDCT for 2d-signals against the scipy.fft.idctn function provided by the popular SciPy library [33]. Similar to the DFT-based approach, the E is prepended to the network N_θ to be verified and allows bound propagation through it by the SoA verifiers for verification of the network against perturbations in the DCT space. The input to the augmented network $N_\theta \circ E$ is a nonuniform interval set in the DCT coefficient space. As with DFT, the DCT coefficients also hold spatial correspondence to certain types of frequencies. This is exploited for attack design in existing works [2, 37] and can similarly be used for deciding $\hat{M}$ in the specification design. As for the magnitude $\hat{\epsilon}$ of ℓ_∞ perturbation of the DCT coefficients, it can be computed according to the theorems presented below. The implementation for defining DCT-based specifications, their verification and certified training is provided in the shared codebase. We repeat the verification experiment done in §4.2 for the DCT-based specifications to observe the verified and adversarial accuracies reported in Fig.9. Notice again from Fig.9 that a) the network CNN7 could be verified as non-trivially robust for specifications allowing only the low frequency perturbations, irrespective of the training approach used to train it, b) the verified accuracy for all networks lower bounds their adversarial accuracy. The outcome a) can be particularly useful as pixel domain-based hurts standard accuracy of the network in return of robustness to all norm-bounded perturbations across the spectrum; therefore, when the latter is not a necessary requirement for a deployment, one can use the proposed approach to certify or train the network for robustness to the specific perturbation spectrum that is expected in that deployment, while incurring less decrease in network's standard accuracy.

Theorem 3 *[$\mathcal{B}_{2,\hat{\epsilon},\hat{M}} \rightarrow \mathcal{B}_{2,\epsilon}$] The output set of IDCT for input set $\mathcal{B}_{2,\hat{\epsilon},\hat{M}}(0)$ is the pixel space ℓ_2 ball $\mathcal{B}_{2,\epsilon}(0) \in \mathbb{R}^D$ with $\epsilon_2 = \hat{\epsilon}_2$, i.e., $\{\bar{\mathcal{F}}^1(x) \mid x \in \mathcal{B}_{2,\epsilon_2,\hat{M}}(0)\} = \mathcal{B}_{2,\epsilon_2}(0)$.*

The result follows from the orthonormality of F:

$$\begin{aligned}\max_{\hat{x}\in\mathcal{B}_{2,\epsilon_2,\hat{M}}(0)} \|\bar{\mathcal{F}}^1(\hat{x})\|_2 &= \max_{\hat{x}\in\mathcal{B}_{2,\epsilon_2,\hat{M}}(0)} \sqrt{(F^T\hat{x})^T(F^T\hat{x})} \\ &= \max_{\hat{x}\in\mathcal{B}_{2,\epsilon_2,\hat{M}}(0)} \sqrt{\hat{x}^T F F^T \hat{x}} \\ &= \max_{\hat{x}\in\mathcal{B}_{2,\epsilon_2,\hat{M}}(0)} \sqrt{\hat{x}^T\hat{x}} = \hat{\epsilon}_2\end{aligned}$$

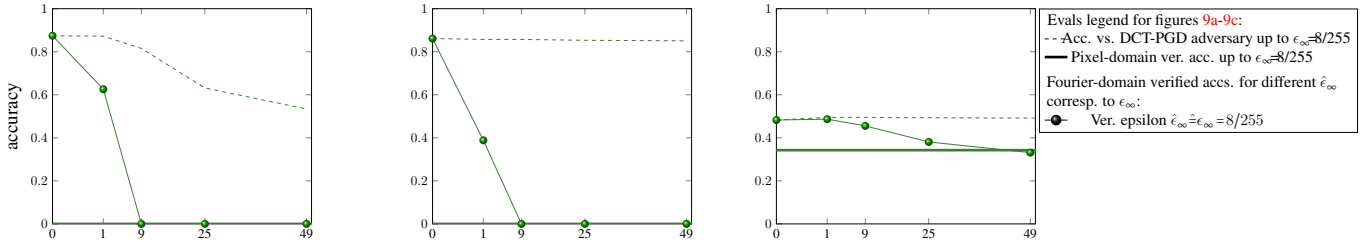
Theorem 4 *[$\mathcal{B}_{\infty,\hat{\epsilon},\hat{M}} \rightarrow \mathcal{B}_{\infty,\epsilon}$] The smallest pixel space ℓ_∞ ball $\mathcal{B}_{\infty,\epsilon}(0)$ enclosing the output set of IDCT for input set $\mathcal{B}_{\infty,\hat{\epsilon},\hat{M}}(0)$ has $\epsilon_\infty = K\hat{\epsilon}_\infty$, where K is the maximal ℓ_1 norm amongst the masked rows of D,*

$$\max_{\hat{x}\in\mathcal{B}_{\infty,\frac{\epsilon_\infty}{K},\hat{M}}(0)} \|F^T\hat{x}\|_\infty = \epsilon_\infty.$$

Let us denote by f^i the i-th row of the IDCT matrix F^T, and by $\tilde{f}^i$ its masked version $\tilde{f}^i = \hat{M}^T \odot f^i$. Proceeding similarly to the proof of theorem 1, we can write:

$$\begin{aligned}\max_{\hat{x}\in\mathcal{B}_{\infty,\frac{\epsilon_\infty}{K},\hat{M}}(0)} \|F^T\hat{x}\|_\infty &= \max_{\hat{x}\in\mathcal{B}_{\infty,\frac{\epsilon_\infty}{K},\hat{M}}(0)} \max_{i\in\{0,\dots,D-1\}} |f^i\hat{x}| \\ &= \max_{i\in\{0,\dots,D-1\}} \max_{\hat{x}\in\mathcal{B}_{\infty,\frac{\epsilon_\infty}{K},\hat{M}}(0)} |f^i\hat{x}| \\ &= \max_{i\in\{0,\dots,D-1\}} \max_{\hat{x}\in\mathcal{B}_{\infty,\frac{\epsilon_\infty}{K}}(0)} \left|f^i\left(\hat{M}\odot\hat{x}\right)\right| \\ &= \max_{i\in\{0,\dots,D-1\}} \max_{\hat{x}\in\mathcal{B}_{\infty,\frac{\epsilon_\infty}{K}}(0)} |\tilde{f}^i\hat{x}| \\ &\leq \max_{i\in\{0,\dots,D-1\}} \max_{\hat{x}\in\mathcal{B}_{\infty,\frac{\epsilon_\infty}{K}}(0)} \sum_{j=0}^{D-1} |\tilde{f}^i[j]||\hat{x}[j]| \\ &= \frac{\epsilon_\infty}{K} \max_{i\in\{0,\dots,D-1\}} \|\tilde{f}^i\|_1 = \epsilon_\infty.\end{aligned}$$

Let us denote by $\text{sign}(a)$ the sign function, and define $z \in_{i\in\{0,\dots,D-1\}} \|\tilde{f}^i\|_1$. Differently from the DFT case, the



(a) Std. training (b) Pixel adv. domain training for $\epsilon_\infty = 8/255$ (c) Pixel domain IBP [24]-based training for $\epsilon_\infty = 8/255$

Figure 9. The figure reports accuracies of differently trained CNN7 networks (7 layers, 62k ReLUs) for additive specification (2). The verified specifications enclose perturbations such that the x-axis specified number of frequency coefficients including and around the central coefficient are allowed to independently vary up to epsilon $\hat{\epsilon}_\infty$. The $\hat{\epsilon}_\infty$ is set corresponding to commonly used pixel-domain ϵ_∞ value of $8/255$ *approximately* as per Theorem 4. Pixel domain verification is done for $\epsilon_\infty = 8/255$. The caption of each figure denotes the approach used to train the network, with d denoting the number of DCT coefficients perturbed during training.

entries of $\hat{x}$ are real-valued. In order to obtain a tight lower bound, we can define $\hat{x}' \in \mathcal{B}_{\infty,\frac{\epsilon_\infty}{K}}(0)$ as a vector whose j-th entry is set as follows: $\hat{x}'[j] = \frac{\epsilon_\infty}{K}\text{sign}(\tilde{f}^z[j])$, resulting in:

$$\begin{aligned}\max_{\hat{x}\in\mathcal{B}_{\infty,\frac{\epsilon_\infty}{K},\hat{M}}(0)} \|F^T\hat{x}\|_\infty &= \max_{i\in\{0,\ldots,D-1\}} \max_{\hat{x}\in\mathcal{B}_{\infty,\frac{\epsilon_\infty}{K}}(0)} |\tilde{f}^i\hat{x}| \\ &\geq \max_{i\in\{0,\ldots,D-1\}} \left|\sum_{j=0}^{D-1} \tilde{f}^i[j]\hat{x}'[j]\right| \\ &= \max_{i\in\{0,\ldots,D-1\}} \frac{\epsilon_\infty}{K}\left|\sum_{j=0}^{D-1} \tilde{f}^i[j]\text{sign}(\tilde{f}^z[j])\right|\end{aligned}$$

For $i = z$:

$$\begin{aligned}\left|\sum_{j=0}^{D-1} \tilde{f}^i[j]\text{sign}(\tilde{f}^z[j])\right| &= \left|\sum_{j=0}^{D-1} |\tilde{f}^z[j]|\right| = \sum_{j=0}^{D-1} |\tilde{f}^z[j]| \\ &= \|\tilde{f}^z\|_1 := \max_{i\in\{0,\ldots,D-1\}} \|\tilde{f}^i\|_1\end{aligned}$$

For $i \in \{0,\ldots,D-1\} \setminus \{z\}$:

$$\begin{aligned}\left|\sum_{j=0}^{D-1} \tilde{f}^i[j]\text{sign}(\tilde{f}^z[j])\right| &\leq \sum_{j=0}^{D-1} |\tilde{f}^i[j]||\text{sign}(\tilde{f}^z[j])| \\ &= \sum_{j=0}^{D-1} |\tilde{f}^i[j]| = \|\tilde{f}^i\|_1 \\ &\leq \max_{i\in\{0,\ldots,D-1\}} \|\tilde{f}^i\|_1\end{aligned}$$

Hence,

$$\begin{aligned}\max_{\hat{x}\in\mathcal{B}_{\infty,\frac{\epsilon_\infty}{K},\hat{M}}(0)} \|F^T\hat{x}\|_\infty &\geq \max_{i\in\{0,\ldots,D-1\}} \frac{\epsilon_\infty}{K}\left|\sum_{j=0}^{D-1} \tilde{f}^i[j]\text{sign}(\tilde{f}^z[j])\right| \\ &= \frac{\epsilon_\infty}{K} \max_{i\in\{0,\ldots,D-1\}} \|\tilde{f}^i\|_1 \\ &= \epsilon_\infty,\end{aligned}$$

concluding the proof. For the orthonormal Type-II IDCT:

$$K = \max_{i\in\{0,\ldots,D-1\}}\left[\left|\frac{\hat{M}[0]}{\sqrt{D}}\right| + \sqrt{\frac{2}{D}}\sum_{j=1}^{n-1}\left|\hat{M}[j]\cos\left(\frac{j\pi}{D}\left(i+\frac{1}{2}\right)\right)\right|\right].$$