A Discussion of Methodological Assumptions

In the main text of the paper our primary methodological assumptions included assumptions on the monotonicity of activation functions, architecture of the neural network, and representation of the input and parameter sets. Below, we discuss how and when these assumptions can be relaxed. We also visualize the various threat models and attack targets in Figure 8.

Activation Monotonicity Activation functions being monotonic is important to the method as input extrema are also output extrema. That is, if the function is monotonically increasing (as is the case with ReLu, Tanh, Sigmoid) then evaluating the activation function at the maximum and minimum values for the domain will correspond to the maximum and minimum values for the range. When an activation is monotonically decreasing, the maximum of the domain interval will correspond to the minimum of the range interval and visa versa. While this makes exposition of the method convenient we note that our propagation method can be extended to activation functions with a bounded number of local maxima and minima. By breaking the function up at these points we get a series of monotonically increasing and decreasing segments and computing the extrema over all such segments suffices for computing the extrema over the range of the function.

Architectural Assumptions In the main text we assume that all layers of the neural network are affine transformations followed by the application of a non-linear activation function. This is generally true of sound certification procedures (certified smoothing not being considered wholly sound, but statistically sound). Our assumption covers convolutional and fully connected layers, but does not handle pooling and batch-norm layers as these layers in their natural formulation are not amenable to reverse propagation with bound propagation methods. This remains an area of study that can enhance bound propagation methods. We are unable to scale to large neural network architectures such as VGG16 due to the fact that current methods for convex relaxation of such complex networks incur too much approximation for non-trivial certification Gowal et al. (2018). As convex relaxation for neural networks advance our presented methodology for certifying explanation robustness will also advance.

Input and Parameter Set Representation In our exposition we only provide a formulation for intervals over inputs and parameter sets. Along with (Gowal et al., 2018), we find that the interval representation of constraints is sufficient. However, there are more complex input properties which may not be captured well by this assumption, and there is research on more expressive abstractions. For a full treatment we reference interested readers to (Gehr et al., 2018). In Gehr et al. (2018) the authors discuss and evalute different approximations for verifying properties of neural networks including the box domain (called intervals in this work), zonotope domain, and the polyhedra domain. In general, more complex abstract domains allow expression of more complex properties.

Bound Propagation Technique In Section 5 we point out that our bounds are not the tightest bounds that can be achieved for the interval domain. There have been many advances in bound propagation for sound certification of neural networks Tjeng et al. (2017); Gehr et al. (2018); Fazlyab et al. (2019); Benussi et al.. More complex propagation techniques arrive at tighter bounds, in our case on the values of δ and E. For example MILP formulations are exact in the limit of computational time and refinement iterations Benussi et al.. However, it is well known that MILP is exponentially more expensive than the methods presented in this paper. The specifications and techniques in this work can be adapted to any solver, and some solvers give tighter bounds at the cost of increased computational complexity. We have chosen IBP for this method as its computational efficiency allows us to scale to CNNs with over half a million parameters which is infeasable with MILP methods (Benussi et al.).

B Effect of Explanation Regularization Methods

In this section we perform an empirical study of the effects of robust explanations training methods, visualized in Figure 5. For this, we revisit the half-moons dataset and we train classifiers with our method, Hessian regularization (Dombrowski et al., 2019), and $L2$ adversarial training (Chen et al., 2019). We find that each of the methods smooth the clssification boundary. We highlight

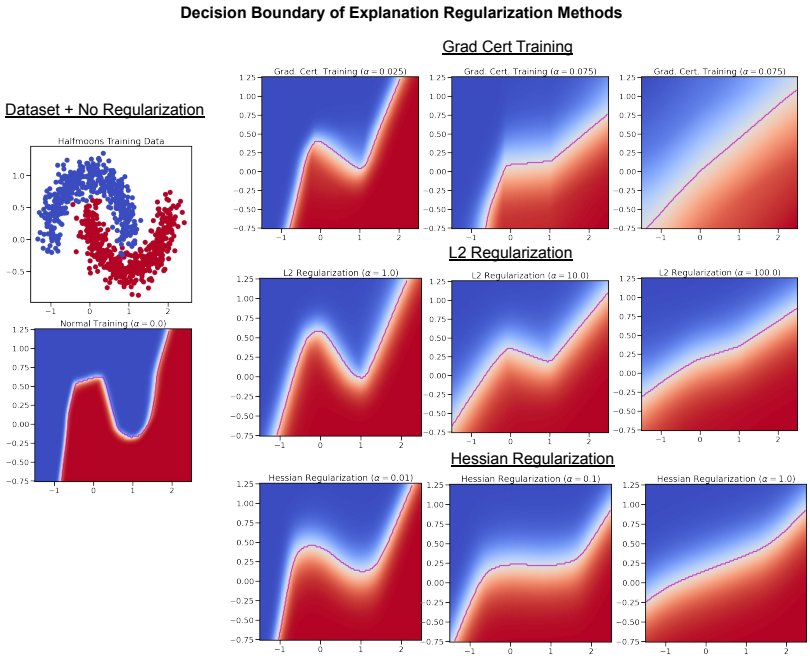

Figure 5: An empirical comparison of how different robust explanation regularization affect the decision boundaries of learned neural networks. From right to left we increase the regularization parameter (denoted α in the main text). **Top Row:** Grad. cert. regularization. **Middle Row:** L2 regularization Chen et al. (2019). **Bottom Row:** Hessian regularization Dombrowski et al. (2019).

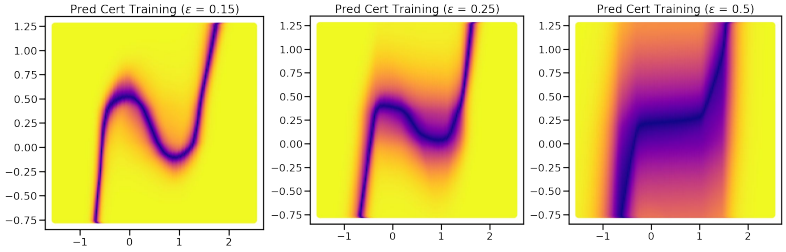

Figure 6: Decision boundaries and confidences for neural networks trained with certified prediction training using (Gowal et al., 2018). We highlight that this method does not have the linearizing effect observed for robust explanations methods and may even lead to misleading explanations when relying on the input gradients as we demonstrate the box in the far right plot.

that while our method and that of (Chen et al., 2019) enforce local linearity, Hessian regularization enforces smoothness but not linearity. We notice that our method is slightly more linear than the other methods for extreme values of the regularization parameters, but in practice, such regularization would not occur due to the considerable performance trade-off that comes with using a linear model. However, we do notice that for more complex datasets the models do have different robustness performance and we find that only our method is able to train NNs with certifiable explanation robustness, see Figure 2.

C Effect of Robust Predictions Training

In this section we empirically compare our training with prediction robustness training. Robust prediction training (Gowal et al., 2018; Mirman et al., 2018) ensures that the prediction does not change for any point in the input interval whereas our method ensures the gradient doesn't change

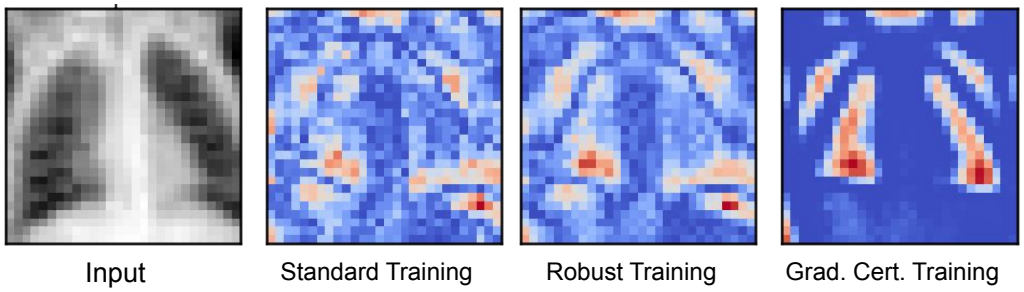

Figure 7: When visualizing the input gradients for different training methods, we see that the prediction robustness training using (Gowal et al., 2018) leads to more sparse explanations (center gradient), but is not nearly as effective as gradient certified training (far right gradient).

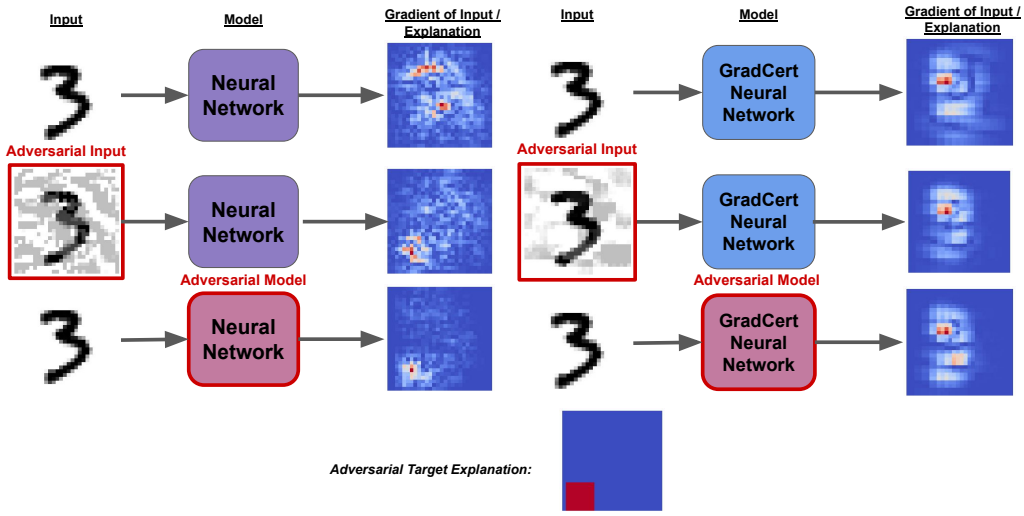

Figure 8: Example of threat models for explanation methods and their affects on a normally trained network versus a network trained with the gradient certification loss. **Top row**: the explanations given for an input classified as a '3' for a normally trained neural network (left) and for the GradCert network (right). **Center row**: The result of using the first-order attack proposed in Dombrowski et al. (2019) on the normal neural network (left) where it is successful and the GradCert (right) where it is unsuccessful. **Bottom row**: The result of using a single-image version of Heo et al. (2019) on a normal neural network (left) and a GradCert neural network (right).

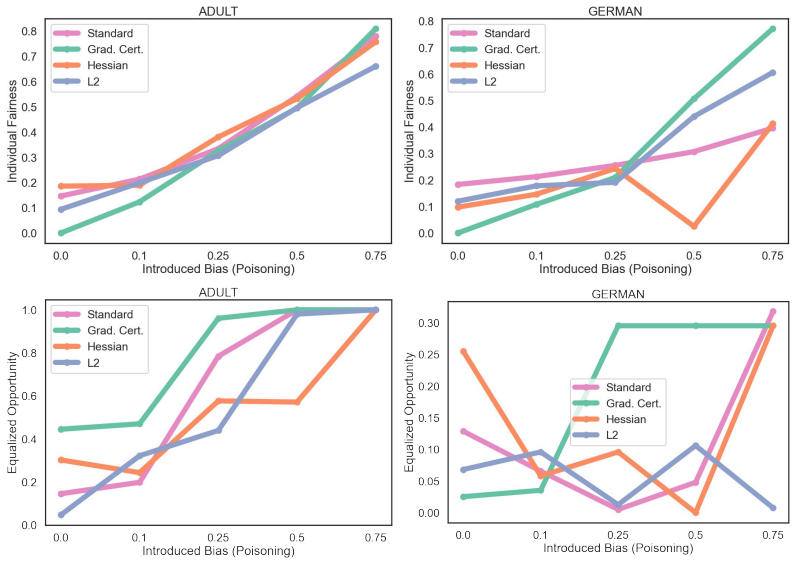

Figure 9: We see that our label poisoning has the intended effect of increasing the bias/unfairness of the classifier. **Top row:** We plot the individual fairness of the classifiers as we increase the poisoning rate. Higher individual fairness values indicate more bias. **Bottom row:** We plot the equalized opportunity, a group fairness metric, and find that for this metric we also increase bias for the Adult dataset, but it is less clear for the German dataset.

inside of the input interval and is unconstrained by the ground truth input. We compare our method with certified robust prediction training proposed in (Gowal et al., 2018). In Figure 6 we plot the resulting neural network decision boundary for different values of α when training with certified prediction robustness. We highlight that for large values of the regularization parameter α the robust prediction method does not induce a linear classifier. We also place a grey box on the far right plot to indicate where the network explanation based on the input gradient may be misleading. We also run the (Gowal et al., 2018) training methodology on PneumoniaMNIST. We plot the resulting input gradients in Figure 7. We find that certified robust training does lead to sparser input gradients compared with standard training, but our training procedure is noticeably sparser. We also evaluate the average value of δ_i over 100 test-set images for each method and find that standard training has an average $\delta = 10.199$, robust prediction training has an average $\delta = 2.035$, and our method has an average $\delta = 0.026$. Thus, while robust training makes explanations considerably more robust, there is still a two orders of magnitude gap between robust explanations training and robust predictions training.

D Further Fairness Discussion

In order to detect bias, we first train classifiers on dataset in which we introduce bias. In order to introduce bias we take a random proportion $p \in [0, 1]$ of individuals from the majority and minority classes and poison their labels, we call the proportion p the 'induced bias'. For the proportion p of individuals in the majority class we set the label to a positive classification and for proportion p of individuals in the minority class we change the label to a negative classification. The key idea here is that the neural network will pick up on the correlation between the majority/minority features and the label and will predict based on the sensitive attribute. In Figure 9 we plot measures of individual fairness Benussi et al. and group fairness Binns (2020). Individual fairness measures the difference in output for individuals who are comparable, in our case, only differ by values of their sensitive output. For individual fairness what is plotted is the difference in softmax classification vectors for individuals (taken from the test set) who are identical save for their sensitive features. Higher values indicate more unfairness. We highlight that increases in discrimination according to individual fairness metrics rise with our increase in induced bias. In the bottom row of Figure 9 we plot a notion of group fairness which measures statistically how similar different majority and

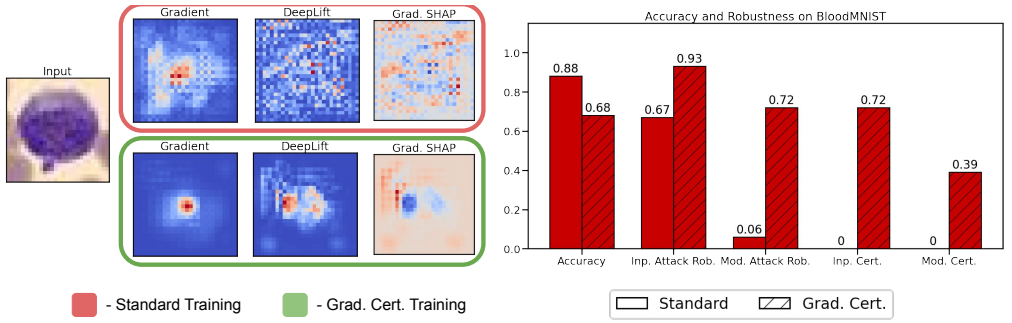

Figure 10: Analysis of three MedMNIST datasets. **Top Row:** Explanations on a test set DermaMNIST image, Grad Cert explanations (outlined in green) are nicely correlated with the key input features. The robustness and accuracy statistics indicate that there is a considerable 20% reduction in test-set accuracy we gain substantial robustness benefits (from 0% to 73% input certified robustness). This figure extends Figure 4 in the main text.

minority groups are treated, again higher values indicate more discrimination. We find that for the Adult dataset, group fairness metrics increase with increases in 'induced bias' but for the German dataset it is less clear of a trend.

We would like to urge all readers that algorithmic fairness is a serious issue. While our method for robust explainability training seems to help indicate when the model is making bias predictions and may supplement fairness analysis, it should not substitute a rigorous fairness evaluation at train time and continuous fairness audits.

E Extended MedMNIST Results

We extend the evaluation of our method on larger scale datasets by exploreing the BloodMNIST dataset which consists of 28 by 28 full-color images of stained blood cells representing 8 different blood disorders. The task of the neural network is to classify these disorders. In Figure 10 we find that we are able to significantly improve both the sparsity of the explanations as well as the well as the robustness of the network; however, it comes at the largest test set accuracy penalty of any dataset tested at 20% test set accuracy loss. We note that further study into hyper-parameter tuning could significantly improve this result.

F Hyper-parameter Values

In this section we report the hyperparameters for the networks trained in the main text. Code to reproduce all of the experiments can be found at *https://github.com/matthewwicker/RobustExplanationConstraintsForNeuralNetworks*.

F.1 Tabular Hyperparameters

For each dataset we use a fully-connected neural network with two layers and 256 hidden neurons per layer. Each network uses ReLU activation functions save for the Hessian training which uses softplus activations. Both our method and the L2 method (Chen et al., 2019) use $\alpha = 1.0$ and Hessian regularization (Dombrowski et al., 2019) uses $\alpha = 0.1$ as using larger values leads to significant performance drops. We see that in Figure 5 that Hessian training is more sensitive to large changes in α. Each network is trained for 100 epochs.

F.2 MNIST Hyperparameters

We split the MNIST hyperparameter section into sections for the fully connected and convolutional networks. As we use the same architecture for the convolutional networks on MedMNIST we report the CNN hyperparameters in the MedMNIST section.

Fully Connected Networks: We use neural networks with two hidden layers and 256 hidden neurons per layer. As before, we use ReLu activation functions. We set $\alpha = 0.5$ for each and vary ϵ_t as shown in Figure 3. Each network is trained for 35 epochs.

F.3 MEDMNIST HYPERPARAMETERS

Below we describe the training parameters used for our MedMNIST experiments.

Convolutional Neural Networks: We consider a small CNN model that was proposed in (Gowal et al., 2018). The network contains of two convolutional layers with 4 by 4 filters. The first layer consists of 16 filters and the second consists of 32 filters. We then flatten the features and pass it to a fully connected hidden layer with 100 neurons. Each layer uses ReLu actiavtions. For MNIST we set $\alpha = 0.5$ for each and vary ϵ_t as shown in Figure 3. Each network is trained for 35 epochs. For MedMNIST we keep $\alpha = 0.5$ and set $\epsilon_t = 0.01$ and $\gamma_t = 0.01$. We found empirically that these gave good results without dropping accuracy.

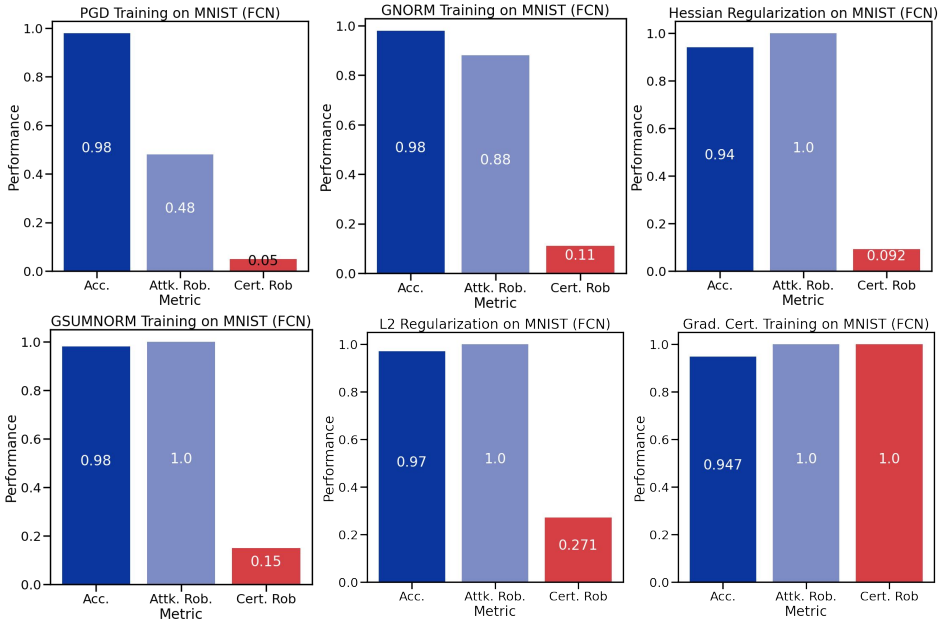

Figure 11: Our method out performs other robust explanation regularization approaches on the MNIST dataset. On the far left we plot standard training, center left we plot Hessian regularization, center right we plot L2 regularization, and far right we plot our method. We find that though regularization methods improve robustness against attacks and even provide some non-trivial certification in the case of L2 regularization, our method considerably out-performs each method with limited accuracy penalty.

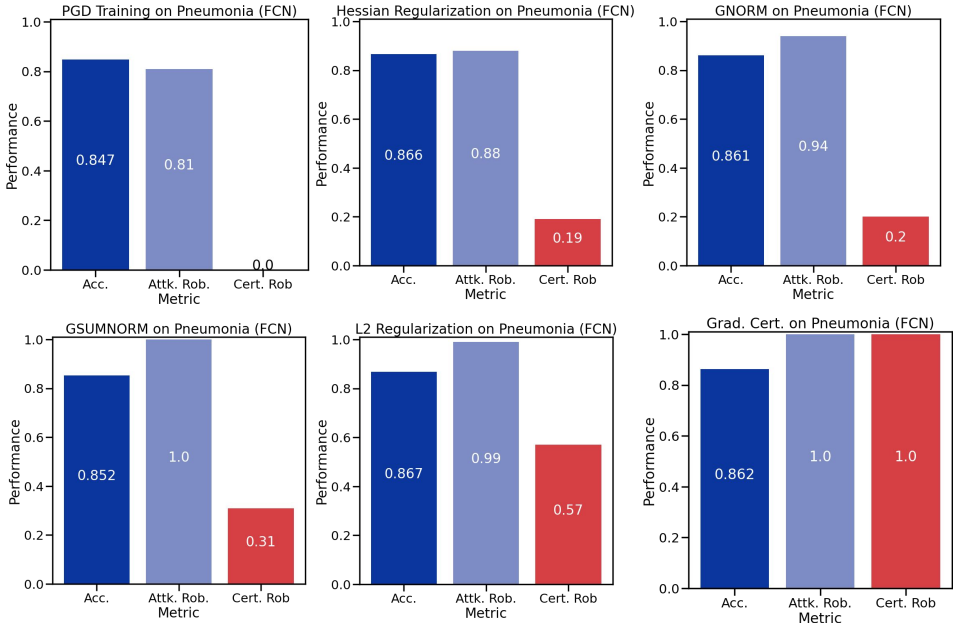

Figure 12: Our method out performs other robust explanation regularization approaches on the Pneumonia MedMNIST dataset. On the far left we plot standard training, center left we plot Hessian regularization, center right we plot L2 regularization, and far right we plot our method. We find that though regularization methods improve robustness against attacks and even provide some non-trivial certification in the case of L2 regularization, our method considerably out-performs each method with limited accuracy penalty.

G Extended Comparisons

In this Section, we provide further experimental comparisons between our method and previous regularization methods. We start by stating the optimization objective of each approach followed by the hyper-parameters used to generate the comparison in Figures 11 and 12. We conclude the section with a discussion of the results. The first regularization tested comes from (Drucker & Le Cun, 1992) and is also discussed in (Chen et al., 2019). The authors propose an objective:

$$\mathcal{L}_{\text{L2}} = \mathcal{L}(f^\theta(x), y) + \alpha ||\nabla_x f^\theta(x')||_2^2$$

where we have added the additional term $x' = x + \mathcal{N}(0, \epsilon)$ and $\epsilon \in \mathbb{R}^n$ defines the width of T. This noise allows us to approximately minimize the gradient magnitude in a ball around the original input. We call this the L2 loss and is what is tested in the main text. All methods trained with this use $\alpha = 0.5$ with $\epsilon = 0.025$ for MNIST and 0.02 for MEDMNIST. In Chen et al. (2019) the authors propose an adversarial version of this loss. In this work, we generalize from integrated gradients to standard gradients and thus drop the 'I' initial in the method name:

$$\mathcal{L}_{\text{GNORM}} = \mathcal{L}(f^\theta(x), y) + \alpha \max_{x' \in T} ||\nabla_x f^\theta(x) - \nabla_{x'} f^\theta(x')||_1$$

it is easy to observe that this is a similar loss to the Gradient Certified loss proposed in this paper save they opt for an ℓ_1 norm penalty and approximately solve the optimization term via stochastic gradient descent whereas the gradient certified loss uses an ℓ_2 norm penalty and computes an over-approximate worst-case solution to the optimization problem. For MNIST we use $\alpha = 1.0$, $\epsilon = 0.025$ and use $\alpha = 0.5$, $\epsilon = 0.02$ for MEDMNIST as these were found to be the best performing parameters. We use 10 iterations of projected gradient descent to solve the minimization term. We also highlight that as this optimization relies on the Hessian we must swap rectified linear units for softplus activations. We also consider Madry et al. (2018) as a baseline method. The robust optimization proposes the following optimization objective:

$$\mathcal{L}_{\text{PGD}} = \max_{x' \in T} \mathcal{L}(f^\theta(x'), y)$$

where the maximization step is solved via projected gradient descent. When training NNs with this loss we again use 10 iterations of PGD with no restarts. We use $\epsilon = 0.025$ for MNIST and $\epsilon = 0.05$ for MEDMNIST. Finally, in (Chen et al., 2019) the authors propose combining the two yielding G-SUM-NORM:

$$\mathcal{L}_{\text{GSUMNORM}} = \max_{x' \in T} \left[\mathcal{L}(f^\theta(x), y) + ||\nabla_x f^\theta(x) - \nabla_{x'} f^\theta(x')||_1 \right]$$

This is combination of robust optimization and robust gradient regularization. For this we use the same parameters as GNORM, i.e., $\alpha = 1.0$, $\epsilon = 0.025$ for MNIST and $\alpha = 0.5$, $\epsilon = 0.02$ for MEDMNIST.

In Figures 11 and 12 we plot test set accuracy, attack robustness, and certified robustness, as defined for explanations in the main text, for each of the above methods. In Figure 11 we train the same one hidden layer neural network with 128 hidden neurons varying only the train time regularization used for each networks. We evaluate the robustness of these networks against an adversary with $\epsilon = 0.0125$ which is considerably lower than what is tested in the main text. This is to display non-trivial behavior for each baseline method. We find that all tested regularization methods lead to a non-trivial level attack robustness. Additionally we find that L2 regularization leads to the best input-certified explanations against an an adversary who perturbs the input by $\epsilon = 0.0125$. We highlight that, as in the main text, Grad. Cert. considerably out-performs all other regularization methods in terms of certified performance. In Figure 12, we again empirically compare the accuracy, attack robustness, and certified robustness of different regularization methods, this time on the pneumonia dataset. We evaluate the network using a smaller ϵ than is considered in the text, here using $\epsilon = 0.02$ with $\gamma = 0.0$ in order to distinguish methods that have comparably less robustness than offered by our method. Our empirical findings in Figure 12 mirror those of Figure 11. In particular, we find that L2 regularization leads to the second best gradient performance while Grad. Cert. continues to out-perform all other methods.

H Certification of Cosine Similarity

In the main text we pose certification for targeted and untargetted attacks through the similarity function h and focused on the ℓ_2-norm as the h function. In this section, we provide the details on the h function and its approximation for when the measure of similarity (or dissimilarity) between two explanations is taken as the cosine similarity.

H.1 Cosine Similarity Bound

We first recall that if $h(v, v')$ is the cosine similarity between the explanations v and v', then the function can be expressed as

$$h(v, v') = \frac{\sum_{i=0}^{n} v'_i v_i}{\sqrt{\sum_{i=0}^{n} v'_i} \sqrt{\sum_{i=0}^{n} v_i}}$$

We can again propose values of v^{cert} that allow us to certify if an adversarial example exists in a given input region.

Targeted Attacks Where v^{targ} is the target explanation vector and $[v^L, v^U]$ is the interval computed by out method, outlined in Section 5, we can compute values for $v' \in [v^L, v^U]$ that over-approximate the how close an adversary can get to v^{targ} w.r.t. the cosine similarity. To minimize the cosine similarity, we would like to maximize the denominator while minimizing the numerator. As we only seek an over-approximate solution, we need not find a single value $v' \in [v^L, v^U]$. Instead we take v^{denom} to be the smallest magnitude value in the range $[v^L, v^U]$ (either 0 or one of the end points) and we take v^{numer} to be the value that minimizes the dot product with the target vector (again, either 0 or one of the end points). The resulting value is an over-approximate minimum for the smallest cosine similarity between the vector v^{targ} and any vector in the interval $[v^L, v^U]$.

I Proof and Theoretical Discussion

In this section of the Appendix we provide a formal proof of Lemma 1 as well as a discussion of its tightness.

I.1 PROOF OF LEMMA 1

Recall that Lemma 1 operates on intervals over matrices. We take the first matrix interval to be $[A^L, A^U]$ and the second matrix interval to be $[B^L, B^U]$. Denoting generic operands in the interval with $A \in \mathbb{R}^{n\times m}$ and $B \in \mathbb{R}^{m\times k}$ we would like to find upper and lower bounds on the product of any two matrices from these intervals (the resulting product being a matrix in $\mathbb{R}^{n\times k}$). For any $i \in [n]$ and $j \in [k]$ the i, j entry in the product-space can be written as $A_{i:} \cdot B_{:j}$, the dot-product of the i^{th} row of A with the j^{th} column of B, which is defined as $\sum_{t=0}^{m} A_{i,t}B_{t,j}$. We now focus bounding the maximum of this dot product (the logic for the minimum follows the same pattern). This is maximized when each term $A_{i,t}B_{t,j}$ is maximized. Given the bi-linear nature of this optimization, the maximum is obtained at one of the four corners of the rectangle $[A^L_{i,t}, A^U_{i,t}] \times [B^L_{t,j}, B^U_{t,j}]$. We now show that Lemma 1 over-approximates the maximizing corner. Recall our upper bound on this rectangle is given as is $A^\mu_{i,t}B^\mu_{t,j} + |A^\mu_{i,t}|B^r_{t,j} + A^r_{i,t}|B^\mu_{t,j}| + |A^r_{i,t}||B^r_{t,j}|$.

In the case that $A^L = A^U$ and $B^L = B^U$, then we have that $A^L = A^U = A^\mu$ and $B^L = B^U = B^\mu$. Thus all of the corners of the rectangle ($[A^L_{i,t}, A^U_{i,t}] \times [B^L_{t,j}, B^U_{t,j}]$) are equal, and the bound returns this value exactly, $A^\mu_{i,t}B^\mu_{t,j}$, as all values super-scripted r are equal to 0.

In the case that $A^L \neq A^U$ but $B^L = B^U$, then we have that $B^L = B^U = B^\mu$, thus the maximum of the rectangle either occurs at $A^L_{i,t}B^\mu_{t,j}$ or $A^U_{i,t}B^\mu_{t,j}$ with the other containing the minimum. The center of the interval by definition is $A^\mu_{i,t}B^\mu_{t,j}$ and we observe that the width of the interval between the maximum and minimum (regardless of which is which) is $2A^r_{i,t}|B^\mu_{t,j}|$, thus the maximum is obtained at $A^\mu_{i,t}B^\mu_{t,j} + A^r|B^\mu|$ and the minimum at $A^\mu_{i,t}B^\mu_{t,j} - A^r|B^\mu|$, as prescribed by the bound (as $B^r_{t,j} = 0$ erasing the contribution of the other terms).

Finally we have the case in which $A^L \neq A^U$ and $B^L \neq B^U$. In this case we have that the maximum and minimum could occur at any one of the four corners of the rectangle. As before, the center of the interval in product space is $A^\mu_{i,t}B^\mu_{t,j}$. And, assuming that $B^\mu_{t,j} \neq 0$, the width contributed by the interval $[A^L_{i,t}, A^U_{i,t}]$ is given by $A^r_{i,t}|B^\mu_{t,j}|$. Equivalently, assuming that $A^\mu_{i,t} \neq 0$, the width contributed by the interval $[B^L_{i,t}, B^U_{i,t}]$ is given by $|A^\mu_{i,t}|B^r_{t,j}$. In the case that both $A^\mu_{i,t} = 0$ and $B^\mu_{t,j} = 0$, the term $A^r_{i,t}B^r_{t,j}$ jointly accounts for both widths exactly. In the case that $A^\mu_{i,t} = 0$ and $B^\mu_{t,j} \neq 0$ our bound introduces approximation error by over counting the width of $[B^U_{t,j}, B^L_{t,j}]$. However this approximation is sound as we have over-approximated the minimum or maximum.

Above we have shown that Lemma 1 is a sounds over-approximation by exhausting all of the possible interval configurations and showing that Lemma 1 is sound for each.

I.2 DISCUSSION OF LEMMA 1 TIGHTNESS

The bound provided in Lemma 1 represents interval bound propagation jointly over two matrices. As an interval bound, it is exact in every case save for when one of the bounds is centered exactly at zero and the other is not. In this case, the alternative (tighter) interval bounding procedure would be to compute each of the four corners $[A^L_{i,t}, A^U_{i,t}] \times [B^L_{t,j}, B^U_{t,j}]$ and subsequently taking the maximum and minimum. This procedure requires an element-wise maximum and minimum operation which make the optimization of such a bound more challenging for auto-differentiation software (Pytorch, Tensorflow, etc), thus Lemma 1 is considerably more desirable when one wants to incorporate our bounds into training. Though at test-time one can gain marginal improvements in certification by taking elementwise maximums and minimums.

J FURTHER RELATED WORK

In our literature review we focus on gradient-based explanations and contextualizing the study of their robustness. Here, we briefly cover explanation methods that use robustness as a primary desiderata. In (Ignatiev et al., 2019) the authors rely on abductive reasoning to get guarantees that are guaranteed to be *minimal* (e.g., in the number of explaining features used) but are not guaranteed to be robust. The authors of (La Malfa et al., 2021) build on abductive-based explanations and consider explanations that are both minimal and optimally robust. In (Blanc et al., 2021) the

authors also consider provable robustness as a key desiderata and achieve this by sampling a black-box model exponentially many times and thus deriving statistical guarantees. Similarly an iterative, greedy strategy is employed by (Ribeiro et al., 2018) to get some statistical guarantees on the quality of their explanations. In counterfactual explanations, (Blanc et al., 2022) theoretically guarantees optimal counterfactuals with their algorithm with analysis of their query complexity. In (Mohammadi et al., 2021), the authors provide provable guarantees in terms of optimal distance, i.e., nearest explanation, and perfect coverage. Concurrently developed with this work, a work was submitted that has a similar aim of providing provable guarantees of counterfactual explanations to changes in the weight-space of a given neural network model Jiang et al. (2023).