Provable Repair of Vision Transformers

Stephanie Nawas[0009−0003−1506−2853], Zhe Tao[0000−0002−4047−699X], and Aditya V. Thakur[0000−0003−3166−1517]

University of California, Davis
{snawas,zhetao,avthakur}@ucdavis.edu

Abstract. Vision Transformers have emerged as state-of-the-art image recognition tools, but may still exhibit incorrect behavior. Incorrect image recognition can have disastrous consequences in safety-critical real-world applications such as self-driving automobiles. In this paper, we present Provable Repair of Vision Transformers (PRoViT), a provable repair approach that guarantees the correct classification of images in a repair set for a given Vision Transformer without modifying its architecture. PRoViT avoids negatively affecting correctly classified images (drawdown) by minimizing the changes made to the Vision Transformer's parameters and original output. We observe that for Vision Transformers, unlike for other architectures such as ResNet or VGG, editing just the parameters in the last layer achieves correctness guarantees and very low drawdown. We introduce a novel method for editing these last-layer parameters that enables PRoViT to efficiently repair state-of-the-art Vision Transformers for thousands of images, far exceeding the capabilities of prior provable repair approaches.

1 Introduction

Vision Transformers [5] have emerged as state-of-the-art image recognition tools, but still exhibit faulty behavior that can result in disastrous real-world consequences. Image recognition plays a significant role in safety-critical applications such as self-driving automobiles [2] and medical diagnosis [14]. Faulty image recognition software has resulted in serious ramifications, including loss of life [8,15]. As Vision Transformers integrate more into real-world applications, it becomes increasingly important to provide guarantees about their correctness to ensure safety.

Recent research on provable repair of deep neural networks (DNNs) [25,26,7,6] explores strategies to provide these guarantees, but such research has not focused on Vision Transformers. In general, provable repair methods guarantee correctness of a DNN's output according to a user-defined repair specification. Provable repair methods strive for the following properties:

- **Efficacy**: The repaired DNN must achieve 100% accuracy on the specified points.
- **Efficiency**: The repair process should be efficient and scale to large DNNs.
- **Low drawdown**: The repair should not negatively affect the previous good behavior of a DNN.
- **High generalization**: The repair should generalize to similar points that are not directly specified in the repair set.

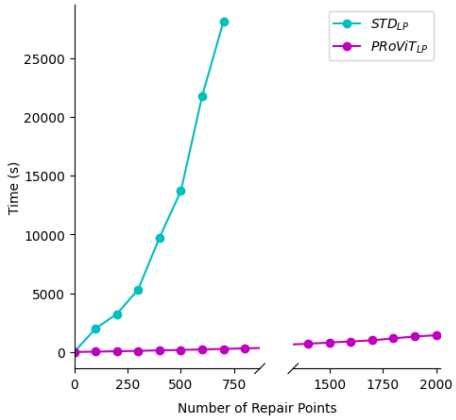

Fig. 1: Runtime comparison of provable repair methods on the Vision Transformer ViT-B/16. After 700 repair points, STD_{LP} runs out of memory, while PRoViT continues to successfully repair thousands of images. See Section 4.2 For more details on this experiment.

These properties set provable repair apart from other DNN editing methods such as retraining and fine tuning. Retraining is inefficient, especially on large models, and the original training set may not be available. Fine tuning has demonstrated a tendency to cause high drawdown [13], meaning the edited DNN has "forgotten" much of its original knowledge. There is an abundance of research on editing pre-trained Transformers to correct faulty behavior [16,17,18,12]. To the best of our knowledge, none of these methods provide provable correctness guarantees.

In this paper, we present Provable Repair of Vision Transformers (PRoViT), a provable repair method that provides correctness guarantees without modifying the original model's architecture. PRoViT is *sound*: a repaired network returned by PRoViT is guaranteed to classify all points in the repair set correctly. Similar to prior provable repair approaches [7,25,26], PRoViT is *not complete*: given a network and a repair specification, it may not find a repaired network. In practice, however, PRoViT successfully repairs the classifications of *thousands of images* on state-of-the-art Vision Transformers. It is efficient and highly scalable; as shown in Figure 1, PRoViT can repair **2000 images in 1423 seconds**. PRoViT also avoids drawdown by minimizing the changes made to the parameters of the Vision Transformer, achieving **0.12% drawdown** when repairing these 2000 images (Table 4).

The **key observation** underlying PRoViT is that, for Vision Transformers, editing the parameters of the last fully-connected linear layer is sufficient to achieve provable repair with low drawdown and high generalization. In particular:

1. We observed that fine tuning just the last layer of the Transformer resulted in efficient repairs with low drawdown and very high generalization. We present the first variant of PRoViT, $\text{PRoViT}_{\text{FT}}$: an approach that **fine tunes the last layer** of the Vision Transformer until all images in the repair set are classified correctly. $\text{PRoViT}_{\text{FT}}$

achieves much higher generalization than the baseline, which fine tuned all layers of the Transformer (54.33% compared to 36.18% in Table 1).
2. We observed that editing the last layer of the Transformer using the standard linear programming (LP) encoding from prior work [26,7] also resulted in repairs with low drawdown. However, this standard LP formulation (STD_{LP}) *does not scale*; the LP solver runs out of memory when trying to repair more than 700 images (Figure 1). Thus, we developed a novel formulation ($\text{PROVIT}_{\text{LP}}$) that scales to *thousands of images*. The **key insight** for $\text{PROVIT}_{\text{LP}}$ is that only the last-layer parameters related to the labels of the images in the repair set need to be modified to achieve provable correctness on the repair set (Section 3.2).
3. We observed that leveraging both last layer fine tuning and $\text{PROVIT}_{\text{LP}}$ to provably repair Vision Transformers allows us to **efficiently achieve both low drawdown and high generalization**. We present a combined approach, $\text{PROVIT}_{\text{FT+LP}}$, to harness the advantages of both fine tuning and $\text{PROVIT}_{\text{LP}}$ (Section 3.3).
4. We observed that last layer provable repair does not work well on non-Vision Transformer architectures (Section 4.3). This suggests that PROVIT is an approach particularly suited for Vision Transformers above all other types of DNNs.

To the best of our knowledge, PROVIT is the only approach for provable repair of Vision Transformers with *all* of the following properties:

- **Provable correctness guarantees**: The repaired network returned by PROVIT is guaranteed to classify all points in the repair set.
- **Transformer architecture-preserving**: The repair does not make any changes to the original architecture of the Vision Transformer.
- **Highly scalable**: PROVIT successfully repairs large Vision Transformers and repair sets with thousands of images.
- **Efficient**: The repair is efficient, remaining within the order of minutes to hours for thousands of points.
- **Low drawdown**: PROVIT does not negatively impact the previous correct classifications of the Vision Transformer after repairing the images in the repair set.
- **High generalization**: The repair generalizes to images beyond those explicitly present in the repair set.

The rest of the paper is organized as follows: Section 2 presents preliminaries; Section 3 presents the PROVIT approach; Section 4 details the experimental evaluation of PROVIT; Section 5 discusses related work; Section 6 concludes.

2 Preliminaries

In this section, we introduce terminology to define provable repair of DNNs (Section 2.1), Vision Transformer architecture (Section 2.2), and the standard last layer LP formulation, STD_{LP} (Section 2.4).

2.1 Provable Repair of Deep Neural Networks

We use $\mathcal{N}^\theta$ to denote a deep neural network (DNN) with parameters θ, and $\mathcal{N}(\boldsymbol{x};\theta) \in \mathbb{R}^n$ to denote the output vector of the DNN on input vector $\boldsymbol{x} \in \mathbb{R}^m$. We drop the parameters θ if they are clear from the context. In this paper, we restrict ourselves to classification tasks; thus, n is the number of labels. We use $accuracy(\mathcal{N}^\theta, \Lambda)$ to represent the accuracy of a DNN $\mathcal{N}^\theta$ on set Λ of inputs and expected labels.

Given a *repair set* $\mathcal{S}$ of inputs and labels, the goal of *architecture-preserving provable repair* is to make small changes to the parameters of a given DNN $\mathcal{N}^\theta$ so that the resulting DNN $\mathcal{N}^{\theta'}$ has 100% accuracy on the repair set $\mathcal{S}$.

Definition 1. *Given a DNN $\mathcal{N}^\theta$ and a* repair set *$\mathcal{S}$ of inputs and labels, an* architecture-preserving provable repair *finds parameters θ' such that $\bigwedge_{(\boldsymbol{x},l)\in\mathcal{S}} \arg\max(\mathcal{N}(\boldsymbol{x};\theta')) = l$; that is, $accuracy(\mathcal{N}^{\theta'}, \mathcal{S}) = 100\%$.*

We use *efficacy* to refer to the accuracy of the repaired network on the given repair set. Apart from efficacy, provable repair methods are also evaluated on *drawdown* and *generalization*.

Definition 2. *A* drawdown set *$\mathcal{D}$ is a set of points disjoint from the repair set and representative of a DNN's existing knowledge. For two DNNs $\mathcal{N}$ and $\mathcal{N}'$, the* drawdown *of $\mathcal{N}'$ with respect to $\mathcal{N}$ is $accuracy(\mathcal{N}, \mathcal{D}) - accuracy(\mathcal{N}', \mathcal{D})$. Lower drawdown is better, representing less knowledge lost during repair.*

Definition 3. *A* generalization set *$\mathcal{G}$ is a set of points disjoint but similar to those in the repair set. For two DNNs $\mathcal{N}$ and $\mathcal{N}'$, the* generalization *of $\mathcal{N}'$ with respect to $\mathcal{N}$ is $accuracy(\mathcal{N}', \mathcal{G}) - accuracy(\mathcal{N}, \mathcal{G})$. Higher generalization is better.*

2.2 Vision Transformers

Our goal is to find an architecture-preserving provable repair approach for Vision Transformers [5]. Vision Transformers are self-attention-based architectures that partition an input image into patches for processing. Each patch is flattened into a 1D vector and passed as input in sequence. Encoder layers process the patches, taking into account their relations to one another via nonlinear operations: within the encoders, there are layers of alternating multiheaded self-attention and multilayer perceptron blocks. Finally, the last encoder layer returns a class token, which is a vector representing the predicted class of the input image. This class token is passed through the final feed-forward layer(s) of the Vision Transformer to determine the label for the input image. For more details on Transformer architectures and Vision Transformers in particular, refer to [28] and [5], respectively.

2.3 Prior Approaches

There are a number of existing provable repair approaches including PRDNN [25], REASSURE [6], MMDNN [7], and APRNN [26]. PRDNN repairs a DNN by translating

its architecture into a "decoupled" DNN, essentially duplicating the network. REASSURE repairs DNNs by adding "patch networks" to the original DNN architecture that edit the behavior of the network in certain input regions. As such, PRDNN and REASSURE are not architecture-preserving, so we turn to MMDNN and APRNN instead.

MMDNN and APRNN are both architecture-preserving provable repair methods, but they cannot repair the encoder layers within Vision Transformers. MMDNN and APRNN can both narrow their focus to only modify the last layer of a model and encode the repair as a linear programming (LP) problem. We refer to the LP encoding for last layer repair as STD_{LP}.

2.4 STD_{LP} Baseline

We now introduce terminology to define the standard last layer LP for provable repair, STD_{LP}. We use $\mathcal{N}^{(:-1)}$ to represent a DNN $\mathcal{N}$ without its last layer $\mathcal{N}^{(-1)}$. Similarly, we use $\theta^{(:-1)}$ to denote $\mathcal{N}$'s parameters excluding those in the last layer $\theta^{(-1)}$. Thus, $\mathcal{N}^{(:-1)}(\boldsymbol{x};\theta^{(:-1)}) \in \mathbb{R}^p$ is the input vector to the last layer of $\mathcal{N}$ for some input vector $\boldsymbol{x}$. p is the size of the input to the last layer. The parameters $\theta^{(-1)} \stackrel{\text{def}}{=} \{\boldsymbol{W}, \boldsymbol{b}\}$ of the last layer consist of weights $\boldsymbol{W}$ and biases $\boldsymbol{b}$. $\boldsymbol{W}$ has shape $p \times n$ and $\boldsymbol{b}$ has shape n, where n is the number of labels in the classification task.

First, we introduce symbolic parameters $\widehat{\theta}^{(-1)} \stackrel{\text{def}}{=} \{\widehat{\boldsymbol{W}}, \widehat{\boldsymbol{b}}\}$ where $\widehat{\boldsymbol{W}}$ is a symbolic matrix corresponding to $\boldsymbol{W}$ and $\widehat{\boldsymbol{b}}$ is a symbolic bias vector corresponding to $\boldsymbol{b}$. $\widehat{\boldsymbol{W}}$ and $\widehat{\boldsymbol{b}}$ represent the new weights and biases we must find to satisfy the repair specification. Now, the output of the layer is $\widehat{\boldsymbol{y}} = \mathcal{N}^{(:-1)}(\boldsymbol{x};\theta^{(:-1)})\widehat{\boldsymbol{W}} + \widehat{\boldsymbol{b}}$ for an input vector $\boldsymbol{x}$. $\widehat{\boldsymbol{y}}$ is a symbolic output vector with n elements, one for each label.

The following formula $\Phi(\boldsymbol{v}, l, \widehat{\boldsymbol{W}}, \widehat{\boldsymbol{b}})$ defines an output vector $\widehat{\boldsymbol{y}}$ by performing the symbolic last linear layer computation and ensures that the argmax of $\widehat{\boldsymbol{y}}$ is l:

$$\Phi(\boldsymbol{v}, l, \widehat{\boldsymbol{W}}, \widehat{\boldsymbol{b}}) \stackrel{\text{def}}{=} \widehat{\boldsymbol{y}} = \boldsymbol{v}\widehat{\boldsymbol{W}} + \widehat{\boldsymbol{b}} \wedge \bigwedge_{i \neq l} \widehat{\boldsymbol{y}}_l > \widehat{\boldsymbol{y}}_i \tag{1}$$

The $\widehat{\boldsymbol{y}} = \boldsymbol{v}\widehat{\boldsymbol{W}} + \widehat{\boldsymbol{b}}$ constraint in the formula defines the output vector $\widehat{\boldsymbol{y}}$ according to the symbolic linear layer computation of multiplying the concrete input vector $\boldsymbol{v}$ with the symbolic weights $\widehat{\boldsymbol{W}}$ and adding the result to the symbolic biases $\widehat{\boldsymbol{b}}$. The rest of the constraints, $\bigwedge_{i \neq l} \widehat{\boldsymbol{y}}_l > \widehat{\boldsymbol{y}}_i$, enforce that the argmax of the output vector $\widehat{\boldsymbol{y}}$ is l. Intuitively, representing the argmax function in the form of linear constraints requires that the value of $\widehat{\boldsymbol{y}}_l$ is greater than all other values $\widehat{\boldsymbol{y}}_i$ in $\widehat{\boldsymbol{y}}$ where $i \neq l$. Using this Φ definition, we formulate STD_{LP} as follows:

$$\begin{aligned} &\min \left\| \boldsymbol{W} - \widehat{\boldsymbol{W}} \right\| + \left\| \boldsymbol{b} - \widehat{\boldsymbol{b}} \right\| \\ \text{s.t.} &\bigwedge_{(\boldsymbol{x}, l) \in \mathcal{S}} \Phi(\boldsymbol{v}, l, \widehat{\boldsymbol{W}}, \widehat{\boldsymbol{b}}) \\ &\text{where } \boldsymbol{v} = \mathcal{N}^{(:-1)}(\boldsymbol{x};\theta^{(:-1)}) \end{aligned} \tag{2}$$

STD_{LP} minimizes the change in weight and bias parameters $\widehat{\boldsymbol{W}}$ and $\widehat{\boldsymbol{b}}$ subject to a set of constraints for each $(\boldsymbol{x}, l)$ pair in the set $\mathcal{S}$. We represent the change in parameters

using the upper bound of the p-norm where $p = 1$ or ∞, which is linear and exact. For each input, Equation 2 applies the Φ formula, imposing that the associated label is the argmax of the associated output vector. The first argument to the Φ formula is the output of the DNN's penultimate layer. In other words, it is the concrete input to the last layer. This collection of linear constraints forms an LP representing the desired behavior of the DNN after solving for the parameters $\widehat{\boldsymbol{W}}$ and $\widehat{\boldsymbol{b}}$.

An off-the-shelf LP solver can find a solution to $\widehat{\boldsymbol{W}}$ and $\widehat{\boldsymbol{b}}$ that satisfies all of the constraints if it exists. We then update the original parameters of the last layer, $\theta^{(-1)}$, with the solution to make $\theta^{(-1)}_{new}$. The repaired $\mathcal{N}$'s parameters are $\theta' = \{\theta^{(:-1)}, \theta^{(-1)}_{new}\}$, meaning that the last layer's parameters are updated while the remaining layers of the DNN are identical to the original.

Proposition 1 ([7]). *Given a repair set $\mathcal{S}$ and DNN $\mathcal{N}$, θ' solved by* $\text{ST}\text{D}_{\text{LP}}$ *(Equation 2) satisfies $\mathcal{S}$.*

Remark 1. The number of variables in $\text{ST}\text{D}_{\text{LP}}$ (Equation 2) is $n \times p + n$ and the number of constraints is $n \times |\mathcal{S}|$.

Proof. The symbolic matrix $\widehat{\boldsymbol{W}}$ is of size $p \times n$, and each element of $\widehat{\boldsymbol{W}}$ is a variable. Similarly, the symbolic vector $\widehat{\boldsymbol{b}}$ is of size n, and each element of $\widehat{\boldsymbol{b}}$ is a variable. In total, there are $n \times p + n$ variables in $\text{ST}\text{D}_{\text{LP}}$.

For each element in the repair set, we add n constraints: one to assign the value of $\boldsymbol{y}$ and $n - 1$ to encode the argmax comparison. Each of these argmax constraints ensures that the symbolic output associated with the correct label is greater than another label. Because there are n labels, there must be $n - 1$ comparison constraints (as there is no need to add a constraint to compare the correct label with itself). There are $|\mathcal{S}|$ points in the repair set, so we multiply $|\mathcal{S}|$ with n to get $n \times |\mathcal{S}|$ total constraints. The constraint encoding the value of $\boldsymbol{y}$ could be omitted by just substituting of the value of $\boldsymbol{y}$ instead, bringing the total number of constraints to $(n - 1) \times |\mathcal{S}|$.

2.5 FT_{all} Baseline

$\text{ST}\text{D}_{\text{LP}}$ is a type of architecture-preserving provable repair approach for Vision Transformers. Consider another approach in this category based on fine-tuning. Fine tuning is a well-studied strategy to adjust a DNN's behavior on a set of points, usually disjoint from the training set. The parameters θ of a DNN $\mathcal{N}$ are updated via gradient descent. We define a variation of fine tuning, FT_{all}, that continues to edit all of the parameters of the DNN until the repair set is satisfied. FT_{all} is not guaranteed to terminate, but if it does, then all inputs are classified correctly; hence, it is a provable repair approach. We will consider FT_{all} and $\text{ST}\text{D}_{\text{LP}}$ baselines to compare against our approach, PRoViT.

3 Approach

This section presents PRoViT, our scalable architecture-preserving provable repair method for Vision Transformers. A key observation in this paper is that editing the parameters of

the last layer of a Vision Transformer is sufficient to find a high-quality provable repair: one with low drawdown and high generalization. There are three variants of PRoViT: $\text{PRoViT}_{\text{FT}}$ (Section 3.1), $\text{PRoViT}_{\text{LP}}$ (Section 3.2), and $\text{PRoViT}_{\text{FT+LP}}$ (Section 3.3).

3.1 Last Layer Fine Tuning: $\text{PRoViT}_{\text{FT}}$

$\text{PRoViT}_{\text{FT}}$ is a gradient descent-based provable repair approach that runs fine tuning on the last layer of the Vision Transformer until all images in the repair set are classified correctly. $\text{PRoViT}_{\text{FT}}$ leverages our observation that editing the last layer of a Vision Transformer leads to high-quality repairs, unlike FT_{all} which edits all weights and biases in the Vision Transformer. $\text{PRoViT}_{\text{FT}}$ is a provable repair approach because it continues to make edits until all images are classified correctly. $\text{PRoViT}_{\text{FT}}$ is efficient and the repair generalizes well to other similar images (up to 54.33% accuracy gained in Experiment 1, Section 4.1).

3.2 Novel LP Formulation: $\text{PRoViT}_{\text{LP}}$

$\text{PRoViT}_{\text{LP}}$ is an LP-based provable repair approach that edits the last layer of the Vision Transformer by solving for the weights and biases. $\text{PRoViT}_{\text{LP}}$ incorporates a novel last layer repair LP formulation that scales significantly better than the baseline STD_{LP} (Section 2.4). The **key insight** behind our scalable LP formulation is that it is sufficient to edit *only* the weights and biases associated with the labels that are present in the repair set to achieve provable correctness. We describe this approach in more detail below:

Let $(\boldsymbol{x}, l)$ be an element of the repair set. It is sufficient to only modify the value of $\boldsymbol{y}_l$ to repair the DNN for $\boldsymbol{x}$. Consequently, it is sufficient to only modify the l-th column of the last-layer weight matrix W. Now let K be the set of labels present in the entire repair set. $\text{PRoViT}_{\text{LP}}$ is based on the observation that it is sufficient *in practice* to only adjust the values of *those particular* $|K|$ elements of $\boldsymbol{y}$ and not modify the rest. Consequently, it is sufficient in practice to only modify the columns corresponding to K of the last-layer weight and bias matrices. See Figure 2 for a visualization of the modifications $\text{PRoViT}_{\text{LP}}$ makes to the Vision Transformer's parameters. In Figure 2, the repair set contains images from just two of the classes, so $|K| = 2$.

Let us now convert K to a subsequence of $[0, 1, \ldots, n-1]$ where n is the total number of labels in the classification task. We define a submatrix of $\boldsymbol{W}$, denoted $\boldsymbol{W}_{:,K}$, and a subvector of $\boldsymbol{b}$, denoted $\boldsymbol{b}_K$. These submatrices are formed by selecting columns of $\boldsymbol{W}$ and $\boldsymbol{b}$ indexed by K. $\boldsymbol{W}_{:,K}$ has shape $p \times |K|$ and $\boldsymbol{b}_K$ has shape $|K|$ where p is the size of the input to the last layer $\mathcal{N}^{(-1)}$.

Example 1. Consider a matrix $\boldsymbol{W} = \begin{bmatrix} 1 & 3 & 5 & 7 & 9 \\ 2 & 4 & 6 & 8 & 10 \end{bmatrix}$. Then $\boldsymbol{W}_{:,Q} = \begin{bmatrix} 3 & 5 & 9 \\ 4 & 6 & 10 \end{bmatrix}$ where $Q = [1, 2, 4]$.

We introduce a symbolic matrix $\widehat{\boldsymbol{W}}_{reduced}$ and a symbolic vector $\widehat{\boldsymbol{b}}_{reduced}$ to represent the weights and biases we must find to satisfy the repair set $\mathcal{S}$. The shapes of $\widehat{\boldsymbol{W}}_{reduced}$ and $\widehat{\boldsymbol{b}}_{reduced}$ match $\boldsymbol{W}_{:,K}$ and $\boldsymbol{b}_K$, respectively, since we will only find new values for the weights and biases associated with the labels in K.

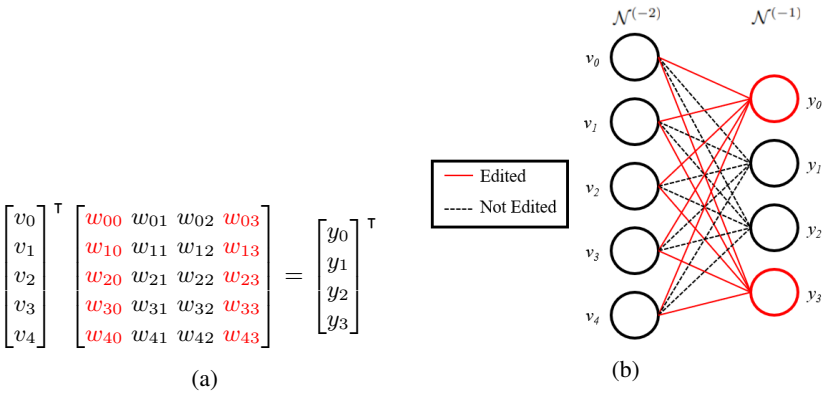

Fig. 2: PRoViT$_{\text{LP}}$ repair visualization of the last layer of a DNN on a 4-label classification task. In this example, the repair set contains images from just two of the classes, so $K = [0, 3]$ and $|K| = 2$. PRoViT$_{\text{LP}}$ only considers the weights in **red** for editing (we omit biases in this example for brevity). In Figure 2b, each input neuron to the last layer is labeled v_0 through v_4 ($\mathcal{N}^{(-2)}$), and each output neuron is labeled y_0 through y_3 ($\mathcal{N}^{(-1)}$). In Figure 2a, the values at those neurons are represented in vector form.

We encode the constraints to ensure that each repair point is correctly classified. Let $\max(Y)$ be a function that returns the maximum value in the vector Y. Φ is defined in Equation 1. Note that in Equation 2, we used Φ to generate constraints associated with all labels in the classification task; in Equation 3 we use Φ to generate constraints associated only with the labels in K. We formulate the reduced LP for PRoViT as follows:

$$\begin{aligned} &\min \left\| \boldsymbol{W}_{:,K} - \widehat{\boldsymbol{W}}_{reduced} \right\| + \left\| \boldsymbol{b}_K - \widehat{\boldsymbol{b}}_{reduced} \right\| \\ &\text{s.t.} \bigwedge_{(\boldsymbol{x},l)\in\mathcal{S}} \left(\Phi\big(\boldsymbol{v}, l, \widehat{\boldsymbol{W}}_{reduced}, \widehat{\boldsymbol{b}}_{reduced}\big) \ \wedge \ \widehat{\boldsymbol{y}}_l > \max\big(\mathcal{N}\left(\boldsymbol{x}, \theta\right)\big) \right) \\ &\qquad \text{where } \boldsymbol{v} = \mathcal{N}^{(:-1)}(\boldsymbol{x}; \theta^{(:-1)}) \end{aligned} \tag{3}$$

An LP solver can find a solution to $\widehat{\boldsymbol{W}}_{reduced}$ and $\widehat{\boldsymbol{b}}_{reduced}$ that satisfies all of the constraints if it exists. We then update the original parameters of the last layer, $\theta^{(-1)}$, with the solution to make $\theta^{(-1)}_{new}$. The repaired $\mathcal{N}$'s parameters are $\theta' = \{\theta^{(:-1)}, \theta^{(-1)}_{new}\}$.

Proposition 2. *Given a repair set $\mathcal{S}$ and DNN $\mathcal{N}$ with parameters θ, the repaired DNN with parameters θ' solved by the LP in Equation 3 satisfies $\mathcal{S}$.*

Proof. Let θ' be the parameters computed by the provable repair technique that uses Equation 3. Let $(\boldsymbol{x}, l)$ be any element of the repair set $\mathcal{S}$, and $\boldsymbol{y}' = \mathcal{N}(\boldsymbol{x}, \theta')$. We will show that $\arg\max(\boldsymbol{y}') = l$.

The argmax constraints in the Φ formula (Equation 1) ensure that

$$\boldsymbol{y}'_l > \boldsymbol{y}'_i \text{ for all } i \in K - \{l\} \tag{4}$$

Let $\boldsymbol{y} = \mathcal{N}(\boldsymbol{x}, \theta)$. Equation 3 includes a constraint that employs the Φ formula to ensure that $\boldsymbol{y}'_l > \boldsymbol{y}_i$ for all $i \in |\boldsymbol{y}|$. The outputs not associated with the labels in K are not modified by the repair; thus, $\boldsymbol{y}'_i = \boldsymbol{y}_i$ for all $i \notin K$. Finally, the $\max(\mathcal{N}(\boldsymbol{x}, \theta))$ constraint ensures that

$$\boldsymbol{y}'_l > \boldsymbol{y}'_i \text{ for all } i \notin K \tag{5}$$

Using Equations 4 and 5, we have $\boldsymbol{y}'_l > \boldsymbol{y}'_i, i \neq l$; that is, $\arg\max(\boldsymbol{y}') = l$.

Remark 2. The number of variables in the LP in Equation 3 is $p \times |K| + |K|$ and the number of constraints is $|\mathcal{S}| \times (|K| + 1)$.

Proof. The size of the symbolic matrix $\widehat{\boldsymbol{W}}_{reduced}$ is $|K| \times p$ and each element of $\widehat{\boldsymbol{W}}_{reduced}$ is a variable. Similarly, the symbolic vector $\widehat{\boldsymbol{b}}_{reduced}$ is of size $|K|$ and each element of $\widehat{\boldsymbol{b}}_{reduced}$ is a variable. In total, there are $p \times |K| + |K|$ variables in the LP.

The $|K| + 1$ constraints consist of $|K| - 1$ constraints to encode the argmax across the labels present in K. There is one constraint added to encode the value of the output $\boldsymbol{y}$ and one more to encode the max function. There are $|K| - 1 + 2 = |K| + 1$ constraints per element in the repair set, so there are $(|K| + 1) \times |\mathcal{S}|$ total constraints. Note that the constraint that encodes the value of $\boldsymbol{y}$ can be omitted in implementations that employ substitution of the value of $\boldsymbol{y}$ instead, which brings the total number of constraints to $|K| \times |\mathcal{S}|$.

As demonstrated in Remark 1 and Remark 2, while the size of the original problem (Equation 2) depends on the total number of labels in the classification task n, the size of the reduced problem (Equation 3) depends on $|K| \leq n$. Figure 2 shows a small-scale instance of how $|K| \ll n$ significantly reduces the number of variables and constraints required for PRoViT$_{\text{LP}}$.

3.3 PRoViT$_{\text{FT+LP}}$

PRoViT$_{\text{FT+LP}}$ is a combination of last-layer fine tuning and PRoViT$_{\text{LP}}$. Given a Vision Transformer $\mathcal{N}$ and a repair set $\mathcal{S}$, PRoViT$_{\text{FT+LP}}$ first runs one iteration of fine tuning on the last layer of $\mathcal{N}$ to quickly gain accuracy on the inputs in $\mathcal{S}$. Fine tuning may achieve 100% efficacy at this stage, in which case PRoViT$_{\text{FT+LP}}$ returns the repaired $\mathcal{N}$. If the efficacy is not 100%, PRoViT$_{\text{FT+LP}}$ runs PRoViT$_{\text{LP}}$ to make additional edits to ensure that all images in $\mathcal{S}$ are classified correctly. As shown in our experimental evaluations, this approach strikes a nice balance between low drawdown and high generalization.

4 Experimental Evaluation

For our experimental evaluation, we repair Vision Transformers trained on ImageNet: ViT-B/16 [5], ViT-L/32 [5] and DeiT [27]. Additionally, we evaluate our approach on ResNet152 [10] and VGG19 [22] to demonstrate that last layer repairs are best suited for Vision Transformers rather than other image recognition architectures. All experiments (except for our scalability study in the Appendix, Section D) were run on a machine with dual 16-core Intel Xeon Silver 4216 CPUs, 384 GB of memory, SSD and

(a) Miniature pinscher with *fog* corruption.

(b) Combination lock with *brightness* corruption.

(c) Traffic light with *frost* corruption.

(d) Street sign with *snow* corruption.

Fig. 3: Examples of images with different corruptions applied (from ImageNet-C [11]).

a NVIDIA RTX A6000 with 48 GB of GPU memory. We implemented PRoViT using PyTorch [20] and Gurobi [9], a mathematical optimization solver for LP problems. Our code is available at `https://github.com/95616ARG/PRoViT`.

We compare the 3 variants of PRoViT: (1) $\textbf{PRoViT}_{\textbf{LP}}$ (abbrev. **LP**), (2) $\textbf{PRoViT}_{\textbf{FT}}$ (abbrev. **FT**), and (3) $\textbf{PRoViT}_{\textbf{FT+LP}}$ (abbrev. **FT+LP**) against the baselines FT_{all} and STD_{LP}. We eliminated other baselines because they do not support Transformer architectures or their approaches were specific to natural language processing tasks. We evaluate the approaches using the following metrics:

- Efficiency: Amount of time taken to achieve 100% accuracy on the repair set.
- Drawdown: Measurement of the loss of accuracy on the test set.
- Generalization: Measurement of the increase in accuracy on images similar to those in the repair set.

In our experimental evaluation, we repair weather-corrupted images from subsets of classes from the ImageNet-C dataset [11]. Figure 3 shows a few examples of ImageNet-C images. This dataset was created by corrupting images from the ImageNet test set. The motivation for this experimental setup is as follows: Consider a self-driving vehicle that processes image data and determines the correct course of action. Suppose that its model classifies street signs and traffic signals poorly in bad weather. We can repair the model using weather-corrupted images from the ImageNet-C dataset with labels "traffic light" and "street sign" (Figure 3c, Figure 3d). Thus, the repair set can be reduced to a specific subset of labels from the original classification task rather than including extra examples from classes on which the model already performs well.

4.1 Experiment 1: Comparison with FT_{all}

In this experiment, we compare FT_{all} with the three variants of PRoViT. The goal of this experiment is to demonstrate the benefits of restricting edits to just the last layer as opposed to editing parameters across all layers of the Vision Transformer.

Repair Sets. We repair weather-corrupted images from the ImageNet-C dataset [11]. We select a random subset K of ImageNet labels such that $|K| < n$ and repair 500 images for each of the selected labels. The 500 images are selected by choosing **4**

Table 1: Drawdown and generalization results in Experiment 1 (Section 4.1). We compare the baseline FT_{all} with the three variants of PRoViT: $\text{PRoViT}_{\text{LP}}$ (LP), $\text{PRoViT}_{\text{FT}}$ (FT), and $\text{PRoViT}_{\text{FT+LP}}$ (FT+LP). $\mathcal{S}$ is the repair set and K is the subset of labels corresponding to the images in $\mathcal{S}$. **Bold number** indicates the *best result*, underlined number indicates the *second best result*, t/o indicates timeout in 20000 seconds. Recall that lower drawdown is better and higher generalization is better. Negative drawdown means that the accuracy on the drawdown set *improved* after repair.

Model	$\lvert K\rvert$	$\lvert\mathcal{S}\rvert$	Drawdown [%] ↓				Generalization [%] ↑			
			FT_{all}	PRoViT			FT_{all}	PRoViT		
				LP	FT	FT+LP		LP	FT	FT+LP
ViT-L/32	4	2000	76.80%	**0.01%**	0.22%	0.08%	24.70%	43.82%	**53.40%**	49.07%
	8	4000	76.77%	**0.01%**	0.67%	0.23%	8.31%	32.18%	**39.74%**	38.96%
	12	6000	76.66%	**0.01%**	1.05%	0.39%	8.35%	35.25%	**42.14%**	41.65%
	16	8000	t/o	**0.01%**	2.19%	0.57%	t/o	32.43%	37.80%	**38.16%**
	20	10000	t/o	**0.02%**	3.24%	0.74%	t/o	31.69%	36.20%	**37.76%**
DeiT	4	2000	81.51%	**-0.01%**	0.28%	**-0.01%**	36.18%	44.76%	**54.33%**	50.01%
	8	4000	81.39%	**-0.01%**	0.80%	0.06%	17.21%	32.87%	**39.34%**	38.05%
	12	6000	t/o	**0.00%**	1.43%	0.14%	t/o	34.77%	**41.74%**	40.37%
	16	8000	t/o	**0.05%**	2.40%	0.40%	t/o	31.99%	37.41%	**38.02%**
	20	10000	t/o	**0.05%**	4.33%	0.62%	t/o	31.45%	35.75%	**37.49%**

corruptions (fog, brightness, frost, and snow) of **5** base images. Each corruption has **5** severity levels. We apply **5** rotations $(-10°, -5°, 0°, 5°, 10°)$ to each image. In total, this creates $4 \times 5 \times 5 \times 5 = 500$ images per label in the repair set. In this experiment, we increase the size of the repair set by incrementing the number of labels we include. Thus, the size of each repair set is $|K| \times 500$. The accuracy of ViT-L/32 on each repair set ranges from 16–21%. The accuracy of DeiT on each repair set ranges from 17–24%.

Drawdown Set. We use the official ILSVRC2012 ImageNet validation set [4] to measure the drawdown of each repair. For ViT-L/32, the top-1 accuracy is 76.972%. For DeiT, the top-1 accuracy is 81.742%.

Generalization Sets. The generalization sets include all weather-corrupted ImageNet-C images within the selected $|K|$ labels that are not present in the corresponding repair set. There are **4** corruptions of the remaining **45** base images. Each corruption has **5** severity levels and we apply the same **5** rotations to each image. So for each label, there are $4 \times 45 \times 5 \times 5 = 4500$ images in the generalization set. In total, the size of each generalization set is $|K| \times 4500$ where $|K|$ is the number of labels in the repair set.

Comparison with Baseline. We compare the performance of $\text{PRoViT}_{\text{FT+LP}}$ against the baseline, FT_{all}. Table 1 shows the drawdown and generalization for both Vision

Table 2: Time spent in Experiment 1 (Section 4.1). We compare the baseline FT_{all} with the three variants of PRoViT: $\text{PRoViT}_{\text{LP}}$ (LP), $\text{PRoViT}_{\text{FT}}$ (FT), and $\text{PRoViT}_{\text{FT+LP}}$ (FT+LP). $\mathcal{S}$ is the repair set and K is the subset of labels corresponding to the images in $\mathcal{S}$. **Bold number** indicates the *best result*, underlined number indicates the *second best result*, t/o indicates timeout in 20000 seconds.

Model	$\lvert K\rvert$	$\lvert\mathcal{S}\rvert$	FT_{all}	PRoViT		
				LP	FT	FT+LP
ViT-L/32	4	2000	8502s	**440s**	537s	541s
	8	4000	12633s	964s	**847s**	1187s
	12	6000	18869s	1598s	**1230s**	1834s
	16	8000	t/o	**2339s**	3225s	2637s
	20	10000	t/o	**3095s**	3960s	3576s
DeiT	4	2000	2681s	**386s**	2282s	535s
	8	4000	9853s	**820s**	1950s	1204s
	12	6000	t/o	**1317s**	1944s	1761s
	16	8000	t/o	**1952s**	2642s	2473s
	20	10000	t/o	**2576s**	3050s	3291s

Transformers. FT_{all} results in terrible drawdown, causing the Vision Transformers to lose most of their original test set accuracy. In addition, $\text{PRoViT}_{\text{FT+LP}}$ consistently outperforms FT_{all}'s generalization by about 20%. $\text{PRoViT}_{\text{FT+LP}}$ also **results in near-zero drawdown**, never reaching more than 1%. Table 2 shows the time comparison between our baseline FT_{all} and our approach $\text{PRoViT}_{\text{FT+LP}}$. FT_{all} takes significantly more time to repair than $\text{PRoViT}_{\text{FT+LP}}$, so we set a timeout of 20000 seconds. Even as repair set sizes reach into the thousands, $\text{PRoViT}_{\text{FT+LP}}$ is more efficient than FT_{all} was on much smaller repair sets.

Ablation Study. We compare the variations of PRoViT, as shown in both Table 1 and Table 2. For ViT-L/32, both the FT and LP variants of PRoViT are faster than $\text{PRoViT}_{\text{FT+LP}}$. However, for DeiT, the LP variant is always the fastest. The $\text{PRoViT}_{\text{LP}}$ approach always achieves the best drawdown. The drawdown is sometimes even negative, meaning that instead of "forgetting" prior knowledge, additional accuracy was gained on the test set. $\text{PRoViT}_{\text{FT}}$ has the best overall generalization. The results for the FT+LP variation of PRoViT provide evidence of a **nice trade-off** between drawdown and generalization. $\text{PRoViT}_{\text{FT+LP}}$ leverages the benefits of both variations while still providing provable correctness guarantees. These results demonstrate PRoViT's efficiency, low drawdown, and high generalization, all while maintaining correctness guarantees on the repair set. PRoViT's success highlights the strength of targeting the last layer of a Vision Transformer for repair.

PRoViT has a lower (better) drawdown than FT_{all}. PRoViT consistently achieves **near 0%** drawdown, while FT_{all}'s drawdown is around 80%. PRoViT has a higher (better) generalization than FT_{all}, and FT_{all} is inefficient and times out on larger repair sets.

4.2 Experiment 2: Comparison with STD_{LP}

In this experiment, we compare our linear programming approach, $\text{PRoViT}_{\text{LP}}$, with the baseline STD_{LP}. This experiment produced the results shown in Figure 1 (Section 1) and is meant to demonstrate the improved performance of our linear programming formulation compared to the standard linear programming approach for provable repair.

Repair Sets. We repair weather-corrupted images from the ImageNet-C dataset [11]. We select a random subset K of ImageNet labels such that $|K| < n$ and repair 50 images for each of the selected labels. The 50 images are selected by choosing **5** base images with the brightness and fog corruptions. Each corruption has **5** severity levels. In total, this creates $5 \times 5 \times 2 = 50$ images per label in the repair set. In this experiment, we increase the size of the repair set by incrementing the number of labels we include. Thus, the size of each repair set is $|K| \times 50$.

Drawdown Set. We use the official ILSVRC2012 ImageNet validation set [4] to measure the drawdown of each repair. For ViT-B/16, the top-1 accuracy is 81.068%.

Generalization Sets. The generalization sets include all of the brightness- and fog-corrupted ImageNet-C images within the selected $|K|$ labels that are not present in the repair set. There are **45** remaining base images. Each corruption has **5** severity levels. So for each label, there are $2 \times 45 \times 5 = 450$ images in the generalization set. In total, the size of each generalization set is $|K| \times 450$.

Results. STD_{LP} runs out of memory when the repair set size is greater than 700 images. Figure 1 shows that $\text{PRoViT}_{\text{LP}}$ can **successfully repair more images** than STD_{LP}. Refer to Table 4 in the Appendix for detailed results of this experiment. In summary, generalization and drawdown results for repair sets for which both $\text{PRoViT}_{\text{LP}}$ and STD_{LP} succeed are quite similar. $\text{PRoViT}_{\text{LP}}$, however, outperforms STD_{LP} in runtime by a significant margin. For the repair set containing 700 images, STD_{LP} takes 28115 seconds whereas $\text{PRoViT}_{\text{LP}}$ only takes 261 seconds (Table 4). This is a **107x speedup**. These results show that PRoViT can efficiently handle much larger repair sets for Vision Transformers. $\text{PRoViT}_{\text{LP}}$ achieves similar drawdown and generalization to STD_{LP} despite having a smaller search space—fewer network parameters available to edit does not decrease the quality of the repair.

$\text{PRoViT}_{\text{LP}}$ makes use of the assumption that, in practice, the repair set will consist of only a small subset of the classes. If the repair set contains images from all classes, $\text{PRoViT}_{\text{LP}}$ reduces to STD_{LP}. We perform additional experiments in the Appendix:

- Section C shows that $\text{PRoViT}_{\text{LP}}$ outperforms STD_{LP} in runtime on *reduced* Vision Transformers. These reduced Vision Transformers only classify a subset of the original 1000 ImageNet labels, and, hence, are easier for STD_{LP} to handle.
- Section D shows that PRoViT is able to handle repair sets with up to 40% of ImageNet classes.

Table 3: Drawdown, generalization and timing results for non-ViT networks Experiment 3 (Section 4.3). This table shows the results for a repair set with 2000 images across 4 different labels. **Bold number** indicates the *best result*, underlined number indicates the *second best result*. See Table 5 in Section B in the Appendix for an extended version of this table.

Model	Drawdown [%]			Generalization [%]			Time [s]		
	PRoViT			PRoViT			PRoViT		
	LP	FT	FT+LP	LP	FT	FT+LP	LP	FT	FT+LP
ResNet152	**11.28%**	77.92%	77.87%	52.90%	**56.58%**	55.94%	1851s	**1323s**	2004s
VGG19	**1.03%**	56.01%	45.97%	50.14%	**52.92%**	52.68%	**621s**	4784s	848s

PRoViT$_{\text{LP}}$ is **orders of magnitude faster** than STD$_{\text{LP}}$, and STD$_{\text{LP}}$ runs out of memory on repair set sizes larger than 700. PRoViT$_{\text{LP}}$ maintains drawdown and generalization parity with STD$_{\text{LP}}$ and continues to achieve low drawdown and good generalization for repair sets on which STD$_{\text{LP}}$ fails.

4.3 Experiment 3: Comparison across Architectures

In this experiment, we demonstrate that restricting the repair to the last layer of a DNN is *not* well suited for other image-recognition architectures, such as ResNet152 [10] and VGG19 [22]. Like Vision Transformers, both ResNet152 and VGG19 have a fully-connected linear last layer which can be repaired with PRoViT for our comparison.

Repair set. The repair set setup for this experiment is identical to that in Experiment 1 (Section 4.1). The accuracy of ResNet152 on the repair sets ranges from 10–12%. The accuracy of VGG19 on the repair sets ranges from 7–9%.

Drawdown set. We use the official ILSVRC2012 ImageNet validation set [4] to measure the drawdown of each repair. For ResNet152, the top-1 accuracy is 78.312%. For VGG19, the top-1 accuracy is 72.376%.

Generalization set. The generalization set setup for this experiment is identical to that in Experiment 1 (Section 4.1).

Results. Table 3 shows the drawdown and generalization of the different variants of PRoViT for the repair set with 2000 images across 4 labels. Table 5 in Section B in the Appendix contains extended results for all repair sets evaluated in Table 1 and Table 2. For these repairs, the generalization is good, but the drawdown is extremely high for both ResNet152 and VGG19. Note that repairing these exact sets of images achieved **near 0%** drawdown on the Vision Transformers. The LP approach resulted in the best drawdown for these networks, but the repairs are still considerably worse than those on the Vision Transformers. This provides insight into the key differences between the ways convolutional architectures like ResNet and VGG distill information within images and how that information is reflected in the final output layer. The theoretical basis

for why PRoViT works well on Vision Transformers but not convolutional networks is left to future work.

5 Related Work

5.1 Formal Methods for Training and Verification

Training DNNs that are robust to adversarial inputs has been extensively researched. Müller et al. [19] present a certified DNN training approach that evaluates worst-case loss on small boxes within the adversarial input region. Balunovic and Vechev [1] propose adversarial training in combination with provable defenses to achieve certified robustness. Their approach aims to strike a balance between high test accuracy and providing robustness certificates.

Formal verification methods prove whether a pre-trained DNN satisfies a given specification. DeepPoly [23] is a verification tool that uses abstract transformers to prove properties of DNNs. DeepT [3] is also a verification tool based on abstract interpretation, specific to Transformer architectures. Both DeepPoly and DeepT return counterexamples to the properties if they are found. These counterexamples can be used as input to PRoViT. Shi et al. [21] have also addressed the verification of Transformers by computing certified bounds to reflect the importance of specified inputs. Their experiments, along with those in DeepT, focus mainly on NLP Transformers as opposed to Vision Transformers.

5.2 Provable Repair of DNNs

The provable repair problem is a related but separate problem for DNNs. Certified training operates as a starting point for correctness guarantees, usually creating a model from scratch. Verification methods aim to produce a certificate of correctness on a pre-trained model; it does not make edits to the DNN at all. Provable repair, on the other hand, provides correctness guarantees for specified inputs by editing the model's parameters.

Provable repair of DNNs was first introduced by Sotoudeh and Thakur [24], and there have been a number of approaches since then. There are two types of provable repair strategies for DNNs: architecture-modifying and architecture-preserving. The first architecture-modifying approach proposed, called PRDNN [25], processes a DNN by decoupling its structure. This architecture modification allows PRDNN to provide correctness guarantees about the parameter edits by formulating the problem as an LP. REASSURE [6] is another architecture-modifying provable repair approach. REASSURE adds small "patch networks" to the original DNN architecture that activate for the inputs that are in the repair set. The parameters of the patch networks can be designed to correct the behavior of the designated points in the repair set. REASSURE does not work on Vision Transformer architectures due to the nonlinear activation functions within the encoder layers.

Architecture-preserving provable repair methods guarantee the correctness of the inputs post-repair without modifying the original structure of the DNN. Goldberger et al. [7] proposed formulating the repair as an SMT query in their approach "minimal

modifications of deep neural networks" (MMDNN), but due to the nonlinear nature of the activation functions, the method does not scale to large DNNs unless the repair is restricted to the last layer. APRNN [26] formulates the repair problem as an LP by adding activation pattern constraints. Thus, APRNN can successfully repair any layer of a DNN. However, neither MMDNN nor APRNN scale to large Vision Transformers, even when restricted to just the last layer, because they consider all parameters corresponding to all output labels during repair. PRoViT is, hence, the only method architecture-preserving provable repair method that scales to Vision Transformers.

5.3 Transformer Editing

While none of the prior provable repair approaches have focused on Transformer architectures, there are many approaches in recent research that focus on editing Transformers without formal correctness guarantees. SERAC [18] tackles the Transformer editing problem by storing edits in an explicit memory, acting as a wrapper around the base Transformer model. In addition to the memory-based cache, SERAC also trains a scope classifier and counterfactual model to determine when to override the base model during inference. Transformer-Patcher [12] is another approach that, like PRoViT, makes edits to the last layer of a Transformer. For each input, however, Transformer-Patcher adds a neuron to the last layer to correct the output. This approach suffers from scalability issues and significantly increases the inference time of the resulting Transformer model.

ROME [16] is another model editing approach for Transformers based on identifying neuron activations that determine a model's predictions. The weights of a Transformer are updated based on these selected neurons to correct a particular "fact" in an NLP Transformer. ROME only has the capability to update one fact at a time, so its scalability is restricted. MEMIT [17] builds on ROME by tracing a "critical path" through the MLP layers and updates the weights along this critical path to allow for thousands of edits at once. This addresses the scalability issue of ROME, however both MEMIT and ROME require the NLP facts to be in the form of a (subject, relation, object) format, and thus are not flexible to other types of model edits. We leave the evaluation of PRoViT on NLP tasks to future work.

6 Conclusion

We presented PRoViT, a scalable architecture-preserving provable repair approach for Vision Transformers. We leveraged the combination of fine tuning and linear programming to make edits to the last layer of Vision Transformers. Our experimental evaluation demonstrates that PRoViT is efficient, generalizes well, and avoids drawdown, all while providing provable correctness guarantees on the repair set.

Acknowledgments We would like to thank the anonymous reviewers for their feedback and suggestions, which greatly improved the quality of the paper. This work is supported in part by NSF grant CCF-2048123 and DOE Award DE-SC0022285.

References

1. Balunovic, M., Vechev, M.: Adversarial training and provable defenses: Bridging the gap. In: International Conference on Learning Representations (2019)
2. Bojarski, M., Testa, D.D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L.D., Monfort, M., Muller, U., Zhang, J., Zhang, X., Zhao, J., Zieba, K.: End to end learning for self-driving cars. CoRR **abs/1604.07316** (2016), http://arxiv.org/abs/1604.07316
3. Bonaert, G., Dimitrov, D.I., Baader, M., Vechev, M.: Fast and precise certification of transformers. In: Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation. pp. 466–481. PLDI 2021, Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3453483.3454056, https://doi.org/10.1145/3453483.3454056
4. Deng, J., Berg, A., Satheesh, S., Su, H., Khosla, A., Li, F.F.: Imagenet large scale visual recognition challenge 2012 (ilsvrc2012) (2012), https://www.image-net.org/challenges/LSVRC/2012/
5. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations (2021), https://openreview.net/forum?id=YicbFdNTTy
6. Fu, F., Li, W.: Sound and complete neural network repair with minimality and locality guarantees. In: International Conference on Learning Representations (2022), https://openreview.net/forum?id=xS8AMYiEav3
7. Goldberger, B., Katz, G., Adi, Y., Keshet, J.: Minimal modifications of deep neural networks using verification. In: LPAR. vol. 2020, p. 23rd (2020)
8. Gonzales, R.: Feds say self-driving uber suv did not recognize jaywalking pedestrian in fatal crash. NPR https://www.npr.org/2019/11/07/777438412/feds-say-self-driving-uber-suv-did-not-recognize-jaywalking-pedestrian-in-fatal- (2019), accessed: 2023-11-16
9. Gurobi Optimization, LLC: Gurobi Optimizer Reference Manual (2023), https://www.gurobi.com
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 770–778 (2016). https://doi.org/10.1109/CVPR.2016.90
11. Hendrycks, D., Dietterich, T.G.: Benchmarking neural network robustness to common corruptions and surface variations. arXiv preprint arXiv:1807.01697 (2018)
12. Huang, Z., Shen, Y., Zhang, X., Zhou, J., Rong, W., Xiong, Z.: Transformer-patcher: One mistake worth one neuron. In: The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023. OpenReview.net (2023), https://openreview.net/pdf?id=4oYUGeGBPm
13. Kemker, R., McClure, M., Abitino, A., Hayes, T.L., Kanan, C.: Measuring catastrophic forgetting in neural networks. In: McIlraith, S.A., Weinberger, K.Q. (eds.) Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative

Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018. pp. 3390–3398. AAAI Press (2018). https://doi.org/10.1609/aaai.v32i1.11651, https://doi.org/10.1609/aaai.v32i1.11651
14. Kermany, D.S., Goldbaum, M., Cai, W., Valentim, C.C., Liang, H., Baxter, S.L., McKeown, A., Yang, G., Wu, X., Yan, F., Dong, J., Prasadha, M.K., Pei, J., Ting, M.Y., Zhu, J., Li, C., Hewett, S., Dong, J., Ziyar, I., Shi, A., Zhang, R., Zheng, L., Hou, R., Shi, W., Fu, X., Duan, Y., Huu, V.A., Wen, C., Zhang, E.D., Zhang, C.L., Li, O., Wang, X., Singer, M.A., Sun, X., Xu, J., Tafreshi, A., Lewis, M.A., Xia, H., Zhang, K.: Identifying medical diagnoses and treatable diseases by image-based deep learning. Cell **172**(5) (2018). https://doi.org/https://doi.org/10.1016/j.cell.2018.02.010, http://www.sciencedirect.com/science/article/pii/S0092867418301545
15. Lee, D.: US opens investigation into Tesla after fatal crash. BBC. https://www.bbc.co.uk/news/technology-36680043 (2016), accessed: 2023-11-16
16. Meng, K., Bau, D., Andonian, A., Belinkov, Y.: Locating and editing factual associations in GPT. In: NeurIPS (2022), http://papers.nips.cc/paper_files/paper/2022/hash/6f1d43d5a82a37e89b0665b33bf3a182-Abstract-Conference.html
17. Meng, K., Sharma, A.S., Andonian, A.J., Belinkov, Y., Bau, D.: Mass-editing memory in a transformer. In: The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023. OpenReview.net (2023), https://openreview.net/pdf?id=MkbcAHIYgyS
18. Mitchell, E., Lin, C., Bosselut, A., Manning, C.D., Finn, C.: Memory-based model editing at scale. In: Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., Sabato, S. (eds.) Proceedings of the 39th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 162, pp. 15817–15831. PMLR (17–23 Jul 2022), https://proceedings.mlr.press/v162/mitchell22a.html
19. Müller, M.N., Eckert, F., Fischer, M., Vechev, M.T.: Certified training: Small boxes are all you need. In: The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023. OpenReview.net (2023), https://openreview.net/pdf?id=7oFuxtJtUMH
20. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E.Z., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library pp. 8024–8035 (2019), https://proceedings.neurips.cc/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html
21. Shi, Z., Zhang, H., Chang, K., Huang, M., Hsieh, C.: Robustness verification for transformers. In: 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020. OpenReview.net (2020), https://openreview.net/forum?id=BJxwPJHFwS
22. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: Bengio, Y., LeCun, Y. (eds.) 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015), http://arxiv.org/abs/1409.1556
23. Singh, G., Gehr, T., Püschel, M., Vechev, M.: An abstract domain for certifying neural networks. Proceedings of the ACM on Programming Languages **3**(POPL), 1–30 (2019)
24. Sotoudeh, M., Thakur, A.V.: Correcting deep neural networks with small, generalizing patches. In: NeurIPS 2019 Workshop on Safety and Robustness in Decision Making (2019)

25. Sotoudeh, M., Thakur, A.V.: Provable repair of deep neural networks. In: 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation (PLDI). ACM (2021). https://doi.org/10.1145/3453483.3454064, https://doi.org/10.1145/3453483.3454064
26. Tao, Z., Nawas, S., Mitchell, J., Thakur, A.V.: Architecture-preserving provable repair of deep neural networks. Proc. ACM Program. Lang. **7**(PLDI) (Jun 2023). https://doi.org/10.1145/3591238, https://doi.org/10.1145/3591238
27. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jegou, H.: Training data-efficient image transformers & distillation through attention. In: Meila, M., Zhang, T. (eds.) Proceedings of the 38th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 139, pp. 10347–10357. PMLR (18–24 Jul 2021), https://proceedings.mlr.press/v139/touvron21a.html
28. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)

A Experiment 2 Results

Table 4: Drawdown, generalization and timing results for Experiment 2 (Section 4.2): repairing ViT-B/16 with STD_{LP} vs. $\text{PROVIT}_{\text{LP}}$. $\mathcal{S}$ is the repair set and K is the subset of labels corresponding to the images in $\mathcal{S}$. **Bold number** indicates the *best result*, "—" indicates that the experiment ran out of memory. Recall that lower drawdown is better and higher generalization is better. Negative drawdown means that the accuracy on the drawdown set *improved* after repair.

$\lvert K\rvert$	$\lvert\mathcal{S}\rvert$	Drawdown [%] ↓		Generalization [%] ↑		Time [s]	
		STD_{LP}	$\text{PROVIT}_{\text{LP}}$	STD_{LP}	$\text{PROVIT}_{\text{LP}}$	STD_{LP}	$\text{PROVIT}_{\text{LP}}$
2	100	0.0%	**-0.002%**	**0.0%**	**0.0%**	1580s	**17s**
4	200	0.004%	**0.002%**	**10.17%**	10.06%	3235s	**62s**
6	300	-0.004%	**-0.01%**	**11.19%**	11.11%	5315s	**94s**
8	400	-0.006%	**-0.01%**	**13.81%**	13.67%	9725s	**139s**
10	500	-0.004%	**-0.008%**	**13.38%**	13.2%	13682s	**173s**
12	600	-0.006%	**-0.01%**	**11.33%**	11.17%	21775s	**217s**
14	700	0.028%	**0.024%**	**12.02%**	11.87%	28115s	**261s**
16	800	—	**0.024%**	—	**11.4%**	—	**304s**
26	1300	—	**0.09%**	—	**11.92%**	—	**627s**
28	1400	—	**0.088%**	—	**12.4%**	—	**697s**
30	1500	—	**0.1%**	—	**12.53%**	—	**806s**
32	1600	—	**0.106%**	—	**12.33%**	—	**894s**
34	1700	—	**0.102%**	—	**12.17%**	—	**989s**
36	1800	—	**0.108%**	—	**11.9%**	—	**1157s**
38	1900	—	**0.114%**	—	**11.82%**	—	**1315s**
40	2000	—	**0.12%**	—	**11.69%**	—	**1423s**

B Experiment 3 Additional Results

Table 5: Drawdown, generalization, and timing results for repairing non-ViT networks: Experiment 3 (Section 4.3). We compare the three variants of PRoViT: $\text{PRoViT}_{\text{LP}}$ (LP), $\text{PRoViT}_{\text{FT}}$ (FT), and $\text{PRoViT}_{\text{FT+LP}}$ (FT+LP). $\mathcal{S}$ is the repair set and K is the subset of labels corresponding to the images in $\mathcal{S}$. **Bold number** indicates the *best result*. Recall that lower drawdown is better and higher generalization is better.

Model	$\lvert K\rvert$	$\lvert\mathcal{S}\rvert$	Drawdown [%] ↓			Generalization [%] ↑			Time [s]		
			LP	FT	FT+LP	LP	FT	FT+LP	LP	FT	FT+LP
ResNet152	4	2000	**11.28%**	77.92%	77.87%	52.90%	**56.58%**	55.94%	1851s	**1323s**	2004s
	8	4000	**13.46%**	77.83%	77.58%	**34.53%**	28.59%	34.24%	3383s	**1246s**	3607s
	12	6000	**18.21%**	77.54%	77.24%	35.88%	33.09%	**36.45%**	5904s	**1314s**	6075s
	16	8000	**21.28%**	77.34%	76.97%	**31.19%**	27.11%	30.74%	8838s	**2104s**	8786s
	20	10000	**20.54%**	77.27%	76.70%	**29.10%**	23.85%	28.00%	12881s	**6614s**	12525s
VGG19	4	2000	**1.03%**	56.01%	45.97%	50.14%	**52.92%**	52.68%	**621s**	4784s	848s
	8	4000	**1.15%**	58.59%	47.74%	28.80%	29.97%	**32.01%**	3138s	**1017s**	3451s
	12	6000	**1.36%**	53.09%	46.28%	28.45%	29.19%	**31.34%**	4526s	**1669s**	4718s
	16	8000	**1.82%**	40.86%	44.35%	24.67%	24.36%	**27.23%**	7218s	**2198s**	7941s
	20	10000	**1.86%**	38.47%	40.51%	22.22%	21.35%	**24.92%**	12835s	**2328s**	11889s

C Experiment 4: Comparison with STD_{LP} on Reduced Vision Transformers

The following experiment contains another baseline comparison with STD_{LP}. Because STD_{LP} does not scale well (as shown in Section 4.2), we have reduced the Vision Transformers to only include the first 50 ImageNet classes instead of the full 1000. This allows us to make a more robust comparison of the two approaches. The experimental setup is as follows:

Repair Sets. We repair weather-corrupted images from the ImageNet-C dataset [11]. We select the first $|K|$ ImageNet labels and repair 100 images for each of the selected labels. The 100 images are selected by choosing **4** corruptions of **5** base images: fog, brightness, frost, and snow. Each corruption has **5** severity levels. In total, this creates $4 \times 5 \times 5 = 100$ images per label in the repair set. In this experiment, we increase the size of the repair set by incrementing the number of labels we include. Thus, the size of each repair set is $|K| \times 500$.

Drawdown Set. We use the official ILSVRC2012 ImageNet validation set [4] to measure the drawdown of each repair. For the reduced ViT-L/32, the top-1 accuracy is 85.52%. For the reduced DeiT, the top-1 accuracy is 88.52%.

Table 6: Drawdown and generalization results in Experiment 4 (Section C): Reduced Vision Transformer repair with just 50 ImageNet classes. We compare the baseline STD_{LP} with the three variants of PRoViT: $\text{PRoViT}_{\text{LP}}$ (LP), $\text{PRoViT}_{\text{FT}}$ (FT), and $\text{PRoViT}_{\text{FT+LP}}$ (FT+LP). $\mathcal{S}$ is the repair set and K is the subset of labels corresponding to the images in $\mathcal{S}$. **Bold number** indicates the *best result*. Recall that lower drawdown is better and higher generalization is better. Negative drawdown means that the accuracy on the drawdown set *improved* after repair.

Model	$\lvert K\rvert$	$\lvert\mathcal{S}\rvert$	Drawdown [%] ↓				Generalization [%] ↑			
			STD_{LP}	PRoViT			STD_{LP}	PRoViT		
				LP	FT	FT+LP		LP	FT	FT+LP
ViT-L/32	4	400	**-0.04%**	**-0.04%**	0.28%	0.04%	**1.80%**	1.48%	1.09%	1.61%
	8	800	**-0.04%**	**-0.04%**	0.20%	0.00%	8.20%	8.15%	**10.47%**	8.50%
	12	1200	0.20%	**0.16%**	2.92%	1.04%	**7.98%**	7.78%	4.84%	6.26%
	16	1600	**0.20%**	0.24%	1.12%	0.28%	5.95%	5.98%	4.86%	**6.14%**
	20	2000	0.20%	**0.04%**	2.60%	1.12%	7.73%	**7.95%**	4.03%	5.93%
DeiT	4	400	-0.04%	-0.04%	**-0.16%**	-0.04%	**0.60%**	**0.60%**	-0.36%	**0.60%**
	8	800	**-0.12%**	0.00%	0.00%	-0.04%	3.35%	4.48%	**6.85%**	5.42%
	12	1200	8.00%	**0.92%**	2.36%	1.36%	10.42%	**11.20%**	7.91%	9.68%
	16	1600	0.12%	**0.08%**	0.52%	0.16%	6.99%	7.23%	3.58%	**7.89%**
	20	2000	0.36%	**0.32%**	2.68%	1.04%	**10.83%**	10.71%	6.36%	8.85%

Generalization Sets. The generalization sets include all weather-corrupted ImageNet-C images within the selected $|K|$ labels that are not present in the repair set. There are **4** corruptions of the remaining **45** base images. Each corruption has **5** severity levels. So for each label, there are $4 \times 45 \times 5 = 900$ images in the generalization set. In total, the size of each generalization set is $|K| \times 900$ where $|K|$ is the number of labels in the repair set.

Results. Table 6 shows that STD_{LP} and PRoViT perform similarly on both drawdown and generalization. Neither method outperforms the other by a significant margin. However, Table 7 shows that PRoViT significantly outperforms STD_{LP} on runtime. This result is similar to the results in Experiment 2, demonstrating consistency across the different comparisons against the baseline. In this experiment, we observe **speedups between 2x and 9x** when comparing PRoViT to STD_{LP}.

Despite reducing the size of the Vision Transformers, $\text{PRoViT}_{\text{LP}}$ still **runs faster** than STD_{LP} while maintaining similar drawdown and generalization.

D Experiment 5: Scalability Study

In this experiment, we explore the scalability of PRoViT for repair sets with a *larger number of labels* in the repair set (larger $|K|$). We run this experiment on a machine with dual 32-core Intel Xeon Platinum 8362 (2.8 GHz) CPUs with 1.5 TB of memory,

Table 7: Time spent in Experiment 4 (Section C): Reduced Vision Transformer repair with just 50 ImageNet classes. We compare the baseline STD_{LP} with the three variants of PRoViT: $\text{PRoViT}_{\text{LP}}$ (LP), $\text{PRoViT}_{\text{FT}}$ (FT), and $\text{PRoViT}_{\text{FT+LP}}$ (FT+LP). **Bold number** indicates the *best result*. $\mathcal{S}$ is the repair set and K is the subset of labels corresponding to the images in $\mathcal{S}$.

Model	$\lvert K\rvert$	$\lvert\mathcal{S}\rvert$	STD_{LP}	PRoViT		
				LP	FT	FT+LP
ViT-L/32	4	400	283s	80s	217s	**79s**
	8	800	1520s	**169s**	418s	170s
	12	1200	1983s	**246s**	614s	**246s**
	16	1600	3309s	346s	1510s	**345s**
	20	2000	3360s	**468s**	1881s	470s
DeiT	4	400	244s	**92s**	228s	**92s**
	8	800	383s	186s	440s	**185s**
	12	1200	1761s	283s	652s	**282s**
	16	1600	1746s	387s	1608s	**386s**
	20	2000	1941s	506s	1998s	**478s**

SSD and NVIDIA H100 with 80 GB of GPU memory. We repair both the ViT-L/32 and DeiT Vision Transformers.

Repair sets. We repair fog-corrupted images from the ImageNet-C dataset [4]. We select $|K|$ random ImageNet labels and repair 5 images with severity level 3 for each of the selected labels. In this experiment, we increase the size of the repair set by incrementing the number of labels we include. Thus, the size of each repair set is $|K| \times 5$. We reduce the number of images included per label compared to prior experiments due to memory constraints—this experiment focuses on PRoViT's scalability as the number of labels increases.

Drawdown set. We use the same drawdown set as in Experiment 1 (Section 4.1). The original accuracies are identical because we are repairing the same Vision Transformers as before.

Generalization sets. The generalization sets include the remaining fog-corrupted images with severity level 3 for each label in the repair set. There are 45 such images per label. Thus, the size of each generalization set is $|K| \times 45$.

Results. Table 8 and Table 9 show that PRoViT succeeds on repair sets that contain up to 400 different labels. 400 labels equates to 40% of the total 1000 ImageNet labels. Similar to the previous experiments, drawdown is lowest on the $\text{PRoViT}_{\text{LP}}$ variation of our approach. Generalization is best when using the $\text{PRoViT}_{\text{FT}}$ and $\text{PRoViT}_{\text{FT+LP}}$ variations. Notably, $\text{PRoViT}_{\text{FT}}$ was the fastest; this is due to the fact that including more labels in the repair set slows down the LP-based approaches. Overall, however,

Table 8: Drawdown and generalization results in Experiment 5 (Section D), which measures scalability to repair sets with a larger number of labels. We compare the three variants of PRoViT: $\text{PRoViT}_{\text{LP}}$ (LP), $\text{PRoViT}_{\text{FT}}$ (FT), and $\text{PRoViT}_{\text{FT+LP}}$ (FT+LP). $\mathcal{S}$ is the repair set and K is the subset of labels corresponding to the images in $\mathcal{S}$. **Bold number** indicates the *best result*, underlined number indicates the *second best result*. Recall that lower drawdown is better and higher generalization is better. Negative drawdown means that the accuracy on the drawdown set *improved* after repair.

Model	% of Labels	$\lvert K\rvert$	$\lvert\mathcal{S}\rvert$	Drawdown [%] ↓			Generalization [%] ↑		
				LP	FT	FT+LP	LP	FT	FT+LP
ViT-L/32	10%	100	500	**0.04%**	1.41%	1.24%	8.63%	21.73%	**21.94%**
	20%	200	1000	**0.06%**	3.37%	3.17%	7.80%	**18.42%**	18.07%
	30%	300	1500	**0.11%**	4.88%	4.52%	6.70%	**14.77%**	14.66%
	40%	400	2000	**0.12%**	5.71%	5.30%	5.43%	**11.49%**	11.38%
DeiT	10%	100	500	**0.01**%	1.24%	1.07%	8.64%	21.76%	**21.82%**
	20%	200	1000	**0.01**%	3.04%	2.67%	8.09%	**17.76%**	17.70%
	30%	300	1500	**0.01**%	4.51%	4.12%	6.94%	14.27%	**14.57%**
	40%	400	2000	**-0.01**%	5.26%	4.88%	5.80%	11.61%	**11.66%**

this experiment demonstrates that PRoViT is **not limited to only repair sets with a minimal number of labels** and can successfully repair thousands of images.

PRoViT can handle repair sets that include up to 400 unique labels while maintaining **near 0%** drawdown and good generalization.

Table 9: Time spent in Experiment 5 (Section D), which measures scalability to repair sets with a larger number of labels. We compare the three variants of PRoViT: $\text{PRoViT}_{\text{LP}}$ (LP), $\text{PRoViT}_{\text{FT}}$ (FT), and $\text{PRoViT}_{\text{FT+LP}}$ (FT+LP). $\mathcal{S}$ is the repair set and K is the subset of labels corresponding to the images in $\mathcal{S}$. **Bold number** indicates the *best result*, underlined number indicates the *second best result*.

Model	$\lvert K\rvert$	$\lvert\mathcal{S}\rvert$	PRoViT		
			LP	FT	FT+LP
ViT-L/32	100	500	3287s	**14s**	1373s
	200	1000	30094s	**28s**	14399s
	300	1500	68888s	**40s**	39554s
	400	2000	151557s	**53s**	111958s
DeiT	100	500	1100s	**10s**	764s
	200	1000	13521s	**19s**	5431s
	300	1500	29769s	**29s**	15225s
	400	2000	85273s	**39s**	41217s