Reducing Certified Regression to Certified Classification for General Poisoning Attacks

Zayd Hammoudeh
University of Oregon
Eugene, OR, USA
zayd@cs.uoregon.edu

Daniel Lowd
University of Oregon
Eugene, OR, USA
lowd@cs.uoregon.edu

***Abstract*—Adversarial training instances can severely distort a model's behavior. This work investigates *certified regression defenses*, which provide guaranteed limits on how much a regressor's prediction may change under a poisoning attack. Our key insight is that certified regression reduces to voting-based certified classification when using median as a model's primary decision function. Coupling our reduction with existing certified classifiers, we propose six new regressors provably-robust to poisoning attacks. To the extent of our knowledge, this is the first work that certifies the robustness of individual regression predictions without any assumptions about the data distribution and model architecture. We also show that the assumptions made by existing state-of-the-art certified classifiers are often overly pessimistic. We introduce a tighter analysis of model robustness, which in many cases results in significantly improved certified guarantees. Lastly, we empirically demonstrate our approaches' effectiveness on both regression and classification data, where the accuracy of up to 50% of test predictions can be guaranteed under 1% training set corruption and up to 30% of predictions under 4% corruption. Our source code is available at https://github.com/ZaydH/certified-regression.**

***Index Terms*—Robust regression, Certified classifier, Data poisoning, Partial set cover, Partial set multicover**

I. Introduction

In a *poisoning attack*, an adversary inserts malicious instances into a model's training set to manipulate one or more target predictions [1, 2, 3, 4, 5]. Kumar et al.'s [6] recent survey of large corporate and governmental organizations found that poisoning attacks were their top ML security concern due to previous successful attacks [7]. Kumar et al. specifically note that most defenses against these attacks lack "fundamental security rigor" and acknowledge that most adversarial ML defenses are like "crypto pre-Shannon" [8].

Kumar et al.'s concerns primarily arise because most ML defenses (including those for poisoning attacks) are *empirical* [9, 10, 11, 12]; such defenses derive from insights into the underlying mechanisms of specific attacks and in turn provide strategies to mitigate the associated vulnerability. The fatal weakness of empirical defenses is that they provide no guarantees of their effectiveness, and attacks can be adapted to bypass them – often with minimal effort [6, 13].

In contrast, *certified defenses* [14, 15, 16, 17] provide a quantifiable guarantee of a prediction's robustness, albeit under specific assumptions. There has been significant recent progress towards lifting the assumptions necessary to certify a classifier's prediction. Today, non-trivial guarantees of the *pointwise robustness* of individual classification predictions are possible without making assumptions about the underlying data distribution or model architecture.

Similar progress has not yet been made for regression. Existing certified regressors generally make strong assumptions about the *data distribution* (e.g., linearity [18], sparsity [19]) that rarely hold in practice. When those assumptions fail to hold, these methods' "guarantees" are not guarantees at all. Another common requirement of existing certified regressors is that the *model architecture* is linear [20], despite other architectures often performing far better [21, 22].

A problem Q is *reducible* to a different problem Q' if an efficient algorithm to solve Q' can also efficiently solve Q [23]. Our *key insight* is that certified regression is reducible to voting-based certified classification. Mapping certified regression to certified classification requires only minimal changes to the certified classifier's architecture, with the robustness certification function identical. Given reducibility, an important takeaway is that certified regression can be viewed as no harder than certified classification.

Coupling our reduction with existing certified classifiers [24, 25, 26], we propose six new certifiably-robust regressors. To the extent of our knowledge, our methods are the first to provide pointwise regression robustness guarantees against poisoning without both distributional and model assumptions.

Our primary contributions are enumerated below.

1) We formalize three paradigms based on median perturbation to map certified regression to certified classification. All of our certified regressors apply one of these paradigms.
2) We propose two provably-robust instance-based regressors – one based on k-nearest neighbors and the other based on all training instances within a feature-space region.
3) We separately propose four ensemble-based certified regressors, where one pair of regressors trains submodels on disjoint data while the other pair allows submodels to be trained on overlapping data.
4) We significantly improve the certification performance of our ensemble-based regressors *and* existing certified classifiers via a tighter analysis of submodel prediction stability.
5) We demonstrate our methods' effectiveness on both regression and classification datasets, where we certify significant fractions of the training set and even outperform state-of-

the-art certified classifiers on binary classification. Note that all proofs are in the supplemental materials.

II. Preliminaries

We begin with a summary of our notation followed by a formalization of our threat model and objective.[1]

Notation Let $[k]$ denote the set of integers $\{1, \ldots, k\}$, and denote the corresponding *power set* $2^{[k]}$. $\mathbb{1}[q]$ is the *indicator function*, which equals 1 if predicate q is true and 0 otherwise. Let $H(k) = \sum_{i=1}^{k} \frac{1}{i}$ denote the *k-th harmonic number*.

In this work, the term "set" refers to a *multiset* where multiple elements may have the same value. Denote set A's *median* $\operatorname{med} A$. In cases where A's cardinality is even, the median is the midpoint between A's $\frac{|A|}{2}$-th and $(\frac{|A|}{2}+1)$-th largest values. Finding the median requires only linear time, meaning median is asymptotically as fast as mean [27].

$x \in \mathcal{X}$ is an *independent variable* (e.g., *feature vector*) of dimension d and $y \in \mathcal{Y} \subseteq \mathbb{R}$ a *response variable* (e.g., *target*). Let $\mathcal{Z} \coloneqq \mathcal{X} \times \mathcal{Y}$ denote the *instance space*. *Training set* $\mathcal{S}$ consists of n training examples. For $m \in \mathbb{N}$ where $m \leq n$, deterministic function $h_{\text{tr}} : \mathcal{Z} \to [m]$ partitions the instance space, and by consequence $\mathcal{S}$. Let $S^{(1)}, \ldots, S^{(m)}$ be the m disjoint training set *blocks* defined by h_{tr} where $\mathcal{S} = \sqcup_{j=1}^{m} S^{(j)}$.

Model $f : \mathcal{X} \to \mathbb{R}$ is trained on full set $\mathcal{S}$. f may be a single *decision function* or the fusion of an *ensemble* of T submodels, where for simplicity T is chosen to be odd. Let f_t denote a submodel where $t \in [T]$. Each f_t has its own training data $\mathcal{D}_t \subset \mathcal{S}$. Submodel training is *deterministic*, meaning training on fixed $\mathcal{D}$ always yields the same model. We separately consider both *partitioned* ($\forall_{t,t'}\, \mathcal{D}_t \cap \mathcal{D}_{t'} = \emptyset$) and *overlapping* ($\exists_{t,t'}\, \mathcal{D}_t \cap \mathcal{D}_{t'} \neq \emptyset$) submodel training data.

Threat Model For arbitrary *test instance* $(x_{\text{te}}, y_{\text{te}})$, the adversary's objective is to alter the model so that the *prediction error* $|f(x_{\text{te}}) - y_{\text{te}}|$ is as large as possible. Our primary threat model considers an adversary that can insert arbitrary instances into training set $\mathcal{S}$ and arbitrarily delete instances from $\mathcal{S}$.[2] The attacker has perfect knowledge of the learner and our method. We make *no assumptions about the underlying data distribution or adversarial training instances*.

Our Objective Determine *certified robustness* R – a guarantee on the number of training instances that can be inserted into or deleted from training set $\mathcal{S}$ without the model prediction ever violating the requirement that $\xi_{\text{l}} \leq f(x_{\text{te}}) \leq \xi_{\text{u}}$, where $\xi_{\text{l}}, \xi_{\text{u}} \in \mathbb{R}$ are user-specified and application dependent. Note that robustness R is *pointwise*, meaning each prediction $f(x_{\text{te}})$ is certified individually.

A. One-Sided vs. Two-Sided Certification Bounds

For simplicity, the remaining sections exclusively describe how to certify a *one-sided upper bound*, $f(x) \leq \xi$, since all other bounds reduce to this base case. For example, certifying a *one-sided lower bound* reduces to certifying an upper bound via negation as $f(x) \geq \xi \Leftrightarrow -f(x) \leq -\xi$. Likewise, a *two-sided bound* is equivalent to the worst one-sided robustness as

$$\begin{aligned} \xi_{\text{l}} \leq f(x) \leq \xi_{\text{u}} &\Leftrightarrow \big(f(x) \geq \xi_{\text{l}}\big) \wedge \big(f(x) \leq \xi_{\text{u}}\big) \\ &\Leftrightarrow \big(-f(x) \leq -\xi_{\text{l}}\big) \wedge \big(f(x) \leq \xi_{\text{u}}\big). \end{aligned} \tag{1}$$

B. Relating Regression and Binary Classification

Binary classification can be viewed as a simple form of regression where $\mathcal{Y} = \{\pm 1\}$. The model's decision function becomes $\operatorname{sgn} f(x_{\text{te}})$ where $\operatorname{sgn} a = +1$ if $a > 0$ and -1 otherwise. While our primary focus is regression, our methods also achieve state-of-the-art results for binary classification.

III. Related Work

Techniques to mitigate (regression) models' implicit fragility have been studied for over half a century. Below we partition these methods into three categories with progressively stronger robustness guarantees.

Resilient Regression Early methods were rooted in *robust statistics* and focused on mitigating the effect of training set outliers. For example, various trimmed loss functions (e.g., Huber [28], Tukey [29]) cap an outlier's influence on a model [30, 31]. Methods like RANSAC [32] employ another common robustness strategy of detecting and removing training set outliers [33, 34].

Adversarially-Robust Regression The above methods primarily target random noise/outliers. Adversarial training instances can be much more insidious since they are crafted to avoid detection by appearing uninfluential and may only affect a very small fraction of test predictions [2, 5]. These factors can combine to make adversarial training instances difficult for resilient methods to fully detect and correct [35].

Some existing adversarial regression defenses do provide pointwise robustness guarantees, albeit under strong assumptions about the underlying data distribution [36]. For example, some work assumes that the training set follows a linear data distribution with arbitrary white, Gaussian noise [18, 37]. Others assume the data distribution's feature matrix is low rank [19]. Conditioning a guarantee on a specific data distribution is inherently precarious – in particular if the strong distributional assumption rarely holds and cannot be easily verified. If the distributional assumption does not hold, any guarantee is no guarantee at all.

Note that there are adversarially-robust regression defenses that provide guarantees without making distributional assumptions [20, 36]. However, their robustness guarantees are themselves distributional. For example, Jagielski et al. [20] bound the clean training data's mean error but provide no pointwise guarantees. In other words, such methods do not provide insight into each prediction's robustness.

A major strength of our approach to certified regression is that we *provide pointwise guarantees without any assumptions*.

Lastly, many existing adversarially-robust regressors consider exclusively linear models [18, 19, 20, 37]. However, state-of-the-art regressors increasingly leverage non-linear

[1] Supplemental Sec. A provides a full nomenclature reference.

[2] Sec. IX considers a somewhat restricted threat model where attackers only make arbitrary deletions but no insertions. This allows us to empirically evaluate our method despite few base models fully utilizing our threat model.

methods [21, 22, 38, 39]. In contrast, our ensemble-based certified regressors support any submodel architecture.

Certified Classification Recent work has proposed numerous *classifiers* provably robust to poisoning and backdoor attacks [15, 16, 17, 40, 41]. These state-of-the-art certified classifiers all rely on *majority voting-based* methods and derive their guarantees by (lower) bounding the number of training set modifications needed for the label with the second-most votes to overtake the plurality label. Jia et al.'s [24] certified k-nearest neighbor (kNN) classifier is the simplest such method, where the set of "votes" is the training labels from the test instance's neighborhood. Due to space, we defer a detailed discussion and analysis of Jia et al. [24]'s method to suppl. Sec. D.

Levine and Feizi [25] propose *deep partition aggregation* (DPA), a general, ensemble-based certified classifier. Suppl. Sec. E describes DPA in detail, but briefly, DPA's deterministic submodels are fully-independent since they are trained on disjoint data. Given a test instance, each submodel predicts a label, and the overall prediction is the ensemble's plurality label. To turn these labels (i.e., votes) into a robustness guarantee, DPA needs to certify each submodel's robustness. However, DPA sidesteps this by always assuming *worst-case* submodel robustness, which we formalize below.

Def. 1. ***Unit-Cost Assumption****: Any modification to a submodel's training set changes the submodel arbitrarily.*

In practice, there are limits to how much a single training set modification will alter a submodel and its predictions – in particular for models with strong inductive biases. Therefore, the unit-cost assumption's pessimism can cause methods like DPA to underestimate a prediction's true robustness. Nonetheless, this assumption greatly simplifies ensemble robustness certification by reducing the task to just submodel vote counting.

Most recently, Wang et al. [26] modify DPA's ensemble so that submodels can be trained on overlapping data, which (slightly) improves the ensemble's certification bounds.

While voting-based methods work well for classification with nominal $\mathcal{Y}$, these ideas have not yet been adapted to regression where $\mathcal{Y}$ is continuous. This work fills in that gap by providing a reduction that adapts certified classifiers to certify regression. We specifically detail the reduction for the certified classifiers proposed by Jia et al. [24], Levine and Feizi [25], and Wang et al. [26]. By building on these methods, we inherit their property of not needing to make assumptions about the data distribution or model architecture.

The fundamental challenge of our reduction is making a continuous output space behave like a robust, nominal label space. We describe the solution to this challenge next.

IV. Warmup: Perturbing a Set's Median

Traditional center statistics such as mean have a *breakdown point* of 0, i.e., altering a single value in a set can shift the mean arbitrarily. In contrast, median has maximum robustness, i.e., a breakdown of 50%. A high breakdown point entails that a statistic is stable and resistant to change. We formalize changes to median below.

Def. 2. ***Median Perturbation****: The task of altering a set's contents so that its median exceeds some specified $\xi \in \mathbb{R}$.*

Throughout this work, determining pointwise robustness R simplifies to quantifying the number of changes that can be made to a set without perturbing its median. To better foster intuitions, we first formalize robustness R w.r.t. simply perturbing a multiset's median and unrelated to any model. Later sections apply these ideas to link certified regression and certified classification.

Formally, let $\mathcal{V}$ be a multiset of cardinality $T \coloneqq |\mathcal{V}|$. Denote the subset of elements in $\mathcal{V}$ that are at most ξ as $\mathcal{V}_\text{l} \coloneqq \{\nu_t \in \mathcal{V} : \nu_t \leq \xi\}$ and denote its complement $\mathcal{V}_\text{u} \coloneqq \mathcal{V} \setminus \mathcal{V}_\text{l}$.

Below we define three different paradigms that constrain how $\mathcal{V}$ is modified. Figure 1 visualizes our first two unweighted paradigms. Note that Fig. 1's values are repeatedly used throughout this paper, including in Fig. 2 for our third median perturbation paradigm and later in Figs. 4 and 5. In all cases below, consider when $\operatorname{med}\mathcal{V} \leq \xi$ since the degenerate case of $\operatorname{med}\mathcal{V} > \xi$ is by definition non-robust.

A. Unweighted Swap Paradigm

Here, set $\mathcal{V}$ has fixed, odd-valued[3] cardinality T. All modifications to $\mathcal{V}$ take the form of "swaps" where a single value in $\mathcal{V}$ is replaced with any real number. Fig. 1b visualizes the unweighted swap paradigm on a simple set $\mathcal{V} = \{2, \ldots, 6\}$ of $T = 5$ values. Lemma 3 tightly bounds the number of arbitrary swaps R that can be made to $\mathcal{V}$ without perturbing its median.

Lemma 3. *For $\xi \in \mathbb{R}$, real multiset $\mathcal{V}$ where $\operatorname{med}\mathcal{V} \leq \xi$ with $T \coloneqq |\mathcal{V}|$ odd, and $\mathcal{V}_\text{l} \coloneqq \{\nu_t \in \mathcal{V} : \nu_t \leq \xi\}$, let $\widetilde{\mathcal{V}}$ be a multiset formed from $\mathcal{V}$ where elements have been arbitrarily replaced. If the number of elements replaced in $\widetilde{\mathcal{V}}$ does not exceed*

$$R = |\mathcal{V}_\text{l}| - \left\lceil \frac{T}{2} \right\rceil, \tag{2}$$

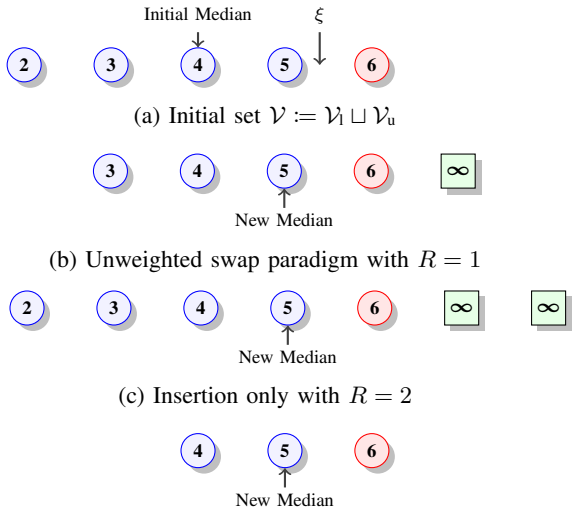
it is guaranteed that $\operatorname{med}\widetilde{\mathcal{V}} \leq \xi$.

Proof sketch. For a set of odd cardinality T, the median is always the set's $\lceil \frac{T}{2} \rceil$-th largest value. For $\mathcal{V}$'s median to be at most ξ, at least $\lceil \frac{T}{2} \rceil$ items in $\mathcal{V}$ cannot exceed ξ. Each swap reduces the number of elements not exceeding ξ by at most one. If there are $|\mathcal{V}_\text{l}|$ elements less than or equal to ξ in $\mathcal{V}$ and there must be at least $\lceil \frac{T}{2} \rceil$ such elements to avoid perturbing the median, then at most $|\mathcal{V}_\text{l}| - \lceil \frac{T}{2} \rceil$ swaps can be performed.

B. Insertion/Deletion Paradigm

For the second paradigm, $\mathcal{V}$ is no longer fixed cardinality (it may expand or contract), and T may be even or odd. Each modification of $\mathcal{V}$ takes the form of either a single deletion or insertion but not both. Figs. 1c and 1d visualize median

[3]Fixing T as odd simplifies the overall formulation and presentation since it ensures that $\mathcal{V}$'s median is always an element in $\mathcal{V}$. In all cases here where T is fixed as odd, T is always a user-selected hyperparameter. Extending our formulation to consider even T is not challenging but is verbose.



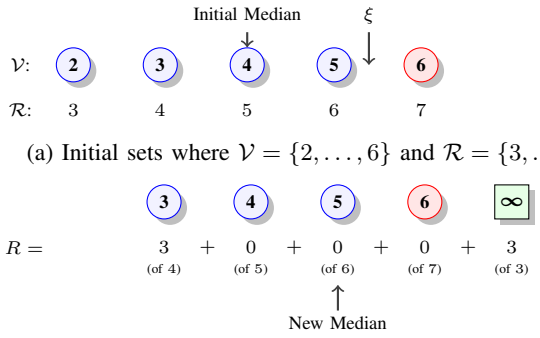
(a) Initial set $\mathcal{V} \coloneqq \mathcal{V}_\text{l} \sqcup \mathcal{V}_\text{u}$

(b) Unweighted swap paradigm with $R = 1$

(c) Insertion only with $R = 2$

(d) Deletion only with $R = 2$

Fig. 1: **Unweighted Median Perturbation**: (1a) Blue denotes elements in subset $\mathcal{V}_\text{l}$, i.e., elements in $\mathcal{V}$ with value at most $\xi = 5.4$. $\mathcal{V}_\text{u}$'s values are red. Each "swap" (1b) switches a value in $\mathcal{V}_\text{l}$ with an arbitrarily large replacement. Deletions (1d) and insertions (1c) are interchangeable (suppl. Lemma 14), with both yielding the same median value in the same number of modifications made to $\mathcal{V}$. In Figs. 1b to 1d above, any additional modifications to the set would perturb the median.



(a) Initial sets where $\mathcal{V} = \{2, \ldots, 6\}$ and $\mathcal{R} = \{3, \ldots, 7\}$

(b) Weighted swap paradigm with $R = 6$

Fig. 2: **Weighted Swap Paradigm**: Extension of Fig. 1 to weighted costs. For simplicity and w.l.o.g., let $\mathcal{R} = \{3, \ldots, 7\}$, i.e., $\forall_t\, r_t = \nu_t + 1$ Fig. 2a is identical to Fig. 1a except below each element ν_t is its corresponding weight r_t. Observe $\Delta = 1$ and $\widetilde{\mathcal{R}}_\text{l} = \{3, 4\}$. Fig. 2b shows that for $R = 6$ (visualized below each element), it is impossible to perturb the median, and any additional weight would be sufficient to swap out $\nu_2 = 3$.

perturbation under insertions and deletions resp. with certified robustness R following Lem. 4. Suppl. Sec. B proves that worst-case insertions and deletions perturb a set's median in exactly the same way and thus are interchangeable. That is why Figs. 1c and 1d have identical certified robustness ($R = 2$).

Lemma 4. *For $\xi \in \mathbb{R}$ and real multiset $\mathcal{V}$ where $\operatorname{med}\mathcal{V} \leq \xi$, define $T \coloneqq |\mathcal{V}|$ and $\mathcal{V}_\text{l} \coloneqq \{\nu_t \in \mathcal{V} : \nu_t \leq \xi\}$. Let $\widetilde{\mathcal{V}}$ be any multiset formed from $\mathcal{V}$ where elements have been arbitrarily deleted and/or inserted. Then, if the total number of inserted and deleted elements in $\widetilde{\mathcal{V}}$ does not exceed*

$$R = 2|\mathcal{V}_\text{l}| - T - 1, \tag{3}$$

it is guaranteed that $\operatorname{med}\widetilde{\mathcal{V}} \leq \xi$.

Eq. (2)'s bound may be non-tight by 1. We did this for consistency with other ideas. See suppl. Sec. G-A for details.

Comparing Eqs. (2) & (3), the insertion/deletion paradigm's robustness R is about twice that of the unweighted swap paradigm. Intuitively, this is because one swap entails two separate operations – both an insertion and a deletion.

C. Weighted Swap Paradigm

The two median-perturbation paradigms above assume that each modification to $\mathcal{V}$ has equivalent cost. Consider a generalized swap paradigm where each value $\nu_t \in \mathcal{V}$ has an associated weight/cost $r_t \in \mathbb{N}$. We seek to tightly bound the budget an attacker could spend with it remaining guaranteed that $f(x_\text{te}) \leq \xi$; we still denote this budget R.

Given $\mathcal{V}_\text{l}$ as above, $\mathcal{R}_\text{l} \coloneqq \{r_t : \nu_t \in \mathcal{V}_\text{l}\}$ contains $\mathcal{V}_\text{l}$'s corresponding weights/costs. Define $\Delta \coloneqq |\mathcal{V}_\text{l}| - \lceil \frac{T}{2} \rceil$, and let multiset $\mathcal{R}_\Delta$ be the Δ smallest values in $\mathcal{R}_\text{l}$ (i.e., $|\mathcal{R}_\Delta| = \Delta$). Directly applying Lem. 3, an obvious but non-optimal bound is

$$R \geq \sum_{r \in \mathcal{R}_\Delta} r. \tag{4}$$

Recall Fig. 1a where $\mathcal{V} = \{2, \ldots, 6\}$ and $\xi = 5.4$. Consider its weighted extension where for simplicity and w.l.o.g. $\mathcal{R} = \{3, \ldots, 7\}$, i.e., $\forall_t\, r_t = \nu_t + 1$. Eq. (4) certifies robustness $R = 3$ for this example. However, Fig. 2b shows $R = 6$ since the budget of the second (i.e., $(\Delta + 1)$-th) largest value in $\mathcal{R}_\text{l}$ can be partially used. Lemma 5 formalizes this insight into a tight bound for median perturbation under weighted swaps.

Lemma 5. *For $\xi \in \mathbb{R}$ and real multiset $\mathcal{V}$ where $\operatorname{med}\mathcal{V} \leq \xi$, let $\mathcal{R}$ be $\mathcal{V}$'s corresponding integral weight multiset where $T \coloneqq |\mathcal{V}| = |\mathcal{R}|$ is fixed and odd. Define $\mathcal{R}_\text{l} \coloneqq \{r_t \in \mathcal{R} : \nu_t \leq \xi\}$, and let $\widetilde{\mathcal{R}}_\text{l}$ be the smallest $(|\mathcal{V}| - \lceil \frac{T}{2} \rceil + 1)$ values in $\mathcal{R}_\text{l}$. Then the cost to perturb $\mathcal{V}$'s median exceeds*

$$R = \sum_{r \in \widetilde{\mathcal{R}}_\text{l}} r - 1. \tag{5}$$

V. Reducing Regression to Voting-Based Binary Classification

We now show how methods used to certify binary classification can be adapted to certify regression. During inference, all voting-based certified methods (both classifiers and regressors) follow the same basic procedure.

First, the model generates a multiset of votes, which for binary classification we denote $\mathcal{V}_{\pm 1}$. Certified classifiers only differ in how $\mathcal{V}_{\pm 1}$ is constructed and in the consequences that construction has on certifying R. For example, $\mathcal{V}_{\pm 1}$ could be

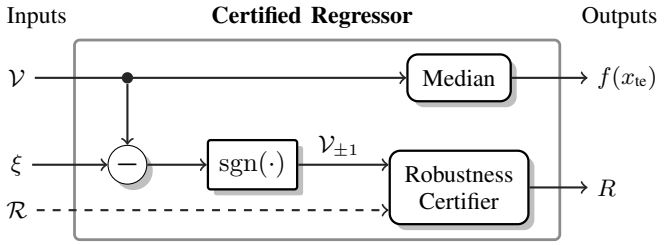


Fig. 3: **Certified Regression to Certified Classification Reduction**: For $x_{\text{te}} \in \mathcal{X}$, the decision function is $f(x_{\text{te}}) \coloneqq \operatorname{med} \mathcal{V}$ – just like voting-based certified classification. Certified regression binarizes $\mathcal{V}$ into $\mathcal{V}_{\pm 1}$, which is used by the robustness certifier (optionally with weights $\mathcal{R}$) to determine R.

a kNN neighborhood or the submodel predictions in an ensemble. Nonetheless, for binary classification, $\mathcal{V}_{\pm 1}$ contains at most two unique values ($+1$ and -1), meaning $\mathcal{V}_{\pm 1}$'s majority label is also its median. In other words, $f(x_{\text{te}}) = \operatorname{med} \mathcal{V}_{\pm 1}$.

To certify robustness R, existing methods rely on a function we term the *robustness certifier*. The function's inputs are votes $\mathcal{V}_{\pm 1}$ and optionally weights/costs $\mathcal{R}$. Implicitly, the certifier knows how the votes were generated and how changes to training set $\mathcal{S}$ could affect $\mathcal{V}_{\pm 1}$. Generally, a simple procedure to construct $\mathcal{V}_{\pm 1}$ entails a simple certifier, and complex construction implies a complex certifier. Fundamentally, for voting-based, binary classification, robustness certification always reduces to the same core idea. If $f(x_{\text{te}}) = \operatorname{med} \mathcal{V}_{\pm 1}$, then for the runner-up label to overtake the majority label, $\mathcal{V}_{\pm 1}$'s median must be perturbed. Therefore, *certifying voting-based, binary classification is simply certifying median perturbation*.

To generalize a voting-based, certified classifier to certify regression, two primary modifications are required; we visualize our regression to classification reduction in Fig. 3.

First, the model is modified from generating binary votes $\mathcal{V}_{\pm 1}$ to generating real-valued ones denoted $\mathcal{V}$. The changes necessary to make this switch are specific to the underlying certified classifier. In some cases, no change is required [24]; for others, ensemble submodel classifiers are simply replaced with submodel regressors [25, 26].

The second modification is more subtle. If $\mathcal{V}$ is real-valued, a robustness certifier expecting binary votes cannot be directly applied. That is where $\xi \in \mathbb{R}$ fits in; it partitions $\mathcal{V}$ into two subsets: $\mathcal{V}_{\text{l}}$ containing all "votes" at most ξ and $\mathcal{V}_{\text{u}}$ containing all "votes" exceeding ξ. We can think of these subsets as two different classes where if $f(x_{\text{te}}) \leq \xi$, $\mathcal{V}_{\text{l}}$ is the majority class and $\mathcal{V}_{\text{u}}$ runner-up. For any prediction $f(x_{\text{te}}) \coloneqq \operatorname{med} \mathcal{V}$, the robustness certifier's output R equals the number of training set modifications that can be made without ever perturbing $\operatorname{med} \mathcal{V}$ beyond ξ.

Lemma 6 formalizes the connection between real-valued and binarized robustness. This symmetry in robustness derives from both tasks' (implicit) shared reliance on median.

Lemma 6. *For $\xi' \in \mathbb{R}$ and real multiset $\mathcal{V}'$ where $\operatorname{med} \mathcal{V}' \leq \xi'$, let $\mathcal{V}_{\pm 1} \coloneqq \{\operatorname{sgn}(\nu_t - \xi') : \nu_t \in \mathcal{V}'\}$. Let $\mathcal{R}$ be the corresponding integral weight multiset of $\mathcal{V}'$ where $|\mathcal{V}'| = |\mathcal{R}|$. Then, under the (un)weighted swap and insertion/deletion paradigms, both $\mathcal{V} = \mathcal{V}'$ with $\xi = \xi'$ and $\mathcal{V} = \mathcal{V}_{\pm 1}$ with $\xi = 0$ have equivalent robustness R.*

By binarizing $\mathcal{V}$, Lem. 6 enables us to directly *reuse robustness certifiers from binary classification to certify regression*.

Our reduction to certified classification entails two primary benefits. First, it allows us to repurpose for regression the diverse set of voting-based, certified classifiers that already exist [24, 25, 26]. Moreover, as new voting-based, certified classifiers are proposed in the future, these yet undiscovered methods can also be reformulated as certified regressors.

Although this work focuses on certified poisoning defenses, other types of certified defenses also rely on voting-based schemes, including randomized smoothing methods for evasion attacks [42, 43]. Our certified regression to certified classification reduction can also be applied to these other types of voting-based defenses as well.

As mentioned above, the procedure to construct the set of votes and to certify robustness is unique to each classifier. The next three sections describe how to certify regression using progressively more complex models, with each method based on a reduction to an existing voting-based certified classifier.

VI. Certified Instance-Based Regression

For the first method, recall from Sec. III that Jia et al. [24] propose a state-of-the-art certified classifier based on kNN. Nearest-neighbor methods are a specific type of *instance-based learner* (IBL), where predictions are made using memorized training instances [44]. IBLs generally rely on the intuition that instances close together in *feature space* ($\mathcal{X}$) have similar target values ($\mathcal{Y}$). Specifically, IBLs search for stored training instances most similar to x_{te} and derive the model prediction from these retrieved *neighbors*.

We partition IBLs into two subcategories:

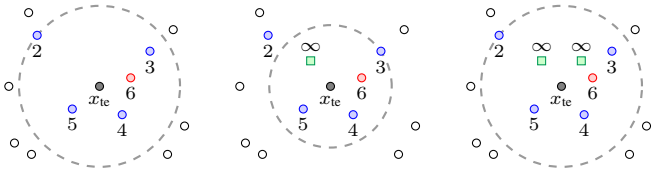
- *Fixed-population neighborhood* methods specify the exact number of "neighbors" when making a prediction.
- *Region-based neighborhood* methods define a neighborhood as all training instances in a specific feature-space region.

These two subcategories calculate certified robustness differently and are discussed separately below.

All IBLs considered here use the same decision rule. Formally, given $x_{\text{te}} \in \mathcal{X}$ and real multiset neighborhood $\mathcal{N}(x_{\text{te}})$ returned by the IBL, the model's prediction is the neighborhood's median, i.e., $f(x_{\text{te}}) \coloneqq \operatorname{med} \mathcal{N}(x_{\text{te}})$. Recall that our goal is to certify that if at most R arbitrary insertions or deletions are made to $\mathcal{S}$, it is guaranteed that $f(x_{\text{te}}) \leq \xi$.

A. Fixed-Population Neighborhood

As the name indicates, fixed-population neighborhood IBLs make predictions using a fixed number of training instances, i.e., $\forall_{x_{\text{te}}} T = |\mathcal{N}(x_{\text{te}})|$. k-nearest neighbors is perhaps the best-known fixed-population method. Traditionally, kNN returns the neighborhood's mean value. For clarity, we will refer to



(a) Unperturbed (b) Fixed-population $R = 1$ (c) Region-based $R = 2$

Fig. 4: **Certified Instance-Based Regression**: Fig. 4a visualizes an unperturbed IBL model. Test instance x_{te}'s neighborhood is visualized as a dashed line with neighborhood $\mathcal{N}(x_{\text{te}})$ identical to $\mathcal{V}$ in Fig. 1a. Fig. 4b shows an attack on a kNN-m model where the neighborhood's cardinality ($T = 5$) is fixed, and the one attack instance (□) replaces one instance in $\mathcal{V}_{\text{l}}$ (○) (source Fig. 1b). A rNN-median model is shown in Fig. 4c, where the two inserted instances (□) do not change the neighborhood's radius (source Fig. 1c).

the version of kNN that uses the neighborhood's median value as *k-Nearest Neighbors Median*, or simply kNN-m.

Our threat model allows the adversary to insert arbitrary training instances and/or delete any existing instances. Fig. 4b visualizes an example attack on a kNN-m regressor. Since k is fixed, inserting a new instance (□) into the neighborhood causes one neighborhood instance to be ejected; in other words, insertions are simply instance swaps. As a worst-case, we assume that the ejected element equals at most threshold ξ, meaning each insertion always maximally increases the neighborhood's median. Under this simplifying assumption, adversarial insertions are always at least as harmful as deletions for fixed-population neighborhood IBLs.

Neighborhood size k is a user-specified hyperparameter so let k be odd-valued. Therefore, these fixed-population neighborhood IBL regressors satisfy all of the criteria of median perturbation under the unweighted swap paradigm where $T = k$. Theorem 7 then follows directly from Lemma 3.

Theorem 7. *Let f be an instance-based regressor trained on set $\mathcal{S}$. Given $\xi \in \mathbb{R}$ and $x_{te} \in \mathcal{X}$, let real multiset $\mathcal{N}(x_{te})$ be x_{te}'s neighborhood under f with fixed, odd-valued cardinality $T \coloneqq |\mathcal{N}(x_{te})|$. Define $\mathcal{V}_{\text{l}} \coloneqq \{y \in \mathcal{N}(x_{te}) : y \leq \xi\}$. Given $f(x_{te}) \coloneqq \operatorname{med} \mathcal{N}(x_{te}) \leq \xi$, then if model f is trained on a modified $\mathcal{S}$ where the total number of inserted and deleted training instances does not exceed*

$$R = |\mathcal{V}_{\text{l}}| - \left\lceil \frac{T}{2} \right\rceil, \tag{6}$$

it is guaranteed that $f(x_{te}) \leq \xi$.

We denote kNN-*m certified regression* as kNN-CR. Due to space, we defer to suppl. Sec. D the proof that when under binary classification, kNN-CR and Jia et al.'s [24] kNN classifier yield identical robustness guarantees.

B. Region-Based Neighborhood

Neighborhood membership does not need to be tied to the number of neighbors. Rather, a neighborhood can be defined by specific criteria, with all stored training instances satisfying those criteria included in the neighborhood. For instance, *radius nearest neighbors* (rNN) defines x_{te}'s neighborhood as all training instances within a given distance of x_{te} [45]. Alternatively, *fully-random decision trees* recursively partition the feature space into disjoint regions, and a neighborhood is defined as all instances within the same feature region [46].

Fig. 4c visualizes an attack on an rNN-median learner, where the adversary inserts malicious instances (□) to perturb the median prediction. Unlike fixed-population neighborhoods, the inserted instances do not cause any existing training instances to be ejected. Rather, inserting and deleting training instances are distinct operations.

It is easy to see that region-based IBLs with median as the decision operator follow Sec. IV-B's insertion/deletion paradigm. Theorem 8 then follows directly from Lemma 4.

Theorem 8. *Let f be an instance-based regressor trained on $\mathcal{S}$ that partitions $\mathcal{X}$ into disjoint regions. Given $x_{te} \in \mathcal{X}$, let real multiset $\mathcal{N}(x_{te})$ be x_{te}'s neighborhood under f where $T \coloneqq |\mathcal{N}(x_{te})|$. For $\xi \in \mathbb{R}$, define $\mathcal{V}_{\text{l}} \coloneqq \{y \in \mathcal{N}(x_{te}) : y \leq \xi\}$. If model f is trained on a modified $\mathcal{S}$ where the total number of inserted and deleted training instances does not exceed*

$$R = 2|\mathcal{V}_{\text{l}}| - T - 1, \tag{7}$$

it is guaranteed that $f(x_{te}) \leq \xi$.

Jia et al. propose an rNN-based certified classifier, with the robustness certifier identical to their kNN method. By using our insertion/deletion paradigm for the robustness certifier instead of Jia et al.'s approach, Eq. (7)'s R roughly doubles.

C. Computational Complexity

Eqs. (6) and (7) require determining $\mathcal{V}_{\text{l}}$'s cardinality, which has complexity $\mathcal{O}(T)$. However, constructing neighborhood $\mathcal{N}(x_{\text{te}})$ can require scanning the entire training set and has complexity $\mathcal{O}(n)$. Therefore, certifying each IBL regression prediction's robustness is in $\mathcal{O}(n)$ – the same as Jia et al.'s certified kNN and rNN classifiers.

VII. Certified Regression for General Models

Instance-based learners lend themselves to robustness certification. However, there are many applications where IBLs perform poorly. This section explores reducing certified regression to a second certified classifier, which will now allow us to use whichever model architecture has the best performance.

Recall that Levine and Feizi's [25] certified classifier, DPA, uses an ensemble trained on partitioned training data. In this section, we first reduce certified regression to certified classification using DPA. We then improve the certification performance of DPA and by extension our certified regressor by using tighter, weighted analysis. All certified regression ensembles we consider have T submodels denoted $f_1, \ldots, f_T$, and the ensemble decision function uses median, i.e., $f(x_{\text{te}}) \coloneqq \operatorname{med}\{f_1(x_{\text{te}}), \ldots, f_T(x_{\text{te}})\}$.

Since ensemble size T is always a user-specified hyperparameter, select odd T. For arbitrary $x_{\text{te}} \in \mathcal{X}$, let

$\mathcal{V} \coloneqq \{f_t(x_{\text{te}}) : t \in [T]\}$. Our goal remains to determine R – a pointwise guarantee on the total number of training set modifications where it remains guaranteed that $f(x_{\text{te}}) \leq \xi$.

A. Partitioned Certified Regression

Here, the T submodel regressors are *fully-independent*, meaning their training sets are disjoint, and each submodel prediction provides no direct insight into any other submodel's behavior. This simple framework makes *no assumptions about the submodel architecture*; the submodels may be non-parametric or parametric, deep or shallow, etc. The only requirement is that each submodel returns a deterministic prediction given its training set and feature vector x_{te}.

Levine and Feizi enforce disjoint submodel training sets by using deterministic function h_{tr} to partition training set $\mathcal{S}$ into T disjoint blocks, $S^{(1)}, \ldots, S^{(T)}$. Formally, for all $t \in [T]$, submodel f_t's training set is $\mathcal{D}_t = S^{(t)}$.

Since each training instance is assigned to exactly one submodel, any training set modification can only affect one submodel. Under the *unit-cost assumption* (Def. 1), each training set modification changes the corresponding submodel's prediction from $f_t(x_{\text{te}})$ to ∞ in the worst case. Thus, perturbing a partitioned ensemble's median prediction follows Sec. IV-A's unweighted swap paradigm where, as explained above, *each perturbed submodel entails one training set modification*.

Via reduction to DPA, Theorem 9 directly applies Lemma 3 to certify unit-cost, partitioned regression's robustness under arbitrary training set insertions and deletions.

Theorem 9. *For $x_{te} \in \mathcal{X}$, $\xi \in \mathbb{R}$, and deterministic function h_{tr} that partitions set $\mathcal{S}$ into disjoint blocks $S^{(1)}, \ldots, S^{(T)}$, let f be an ensemble of T submodels where T is odd, and each deterministic submodel f_t is trained on block $S^{(t)}$. Define $\mathcal{V}_1 \coloneqq \{f_t(x_{te}) : f_t(x_{te}) \leq \xi\}$. Given $f(x_{te}) \coloneqq \text{med}\{f_1(x_{te}), \ldots, f_T(x_{te})\} \leq \xi$, if model f is trained on a modified $\mathcal{S}$ where the total number of inserted and deleted training instances does not exceed*

$$R = |\mathcal{V}_1| - \left\lceil \frac{T}{2} \right\rceil, \tag{8}$$

it is guaranteed that $f(x_{te}) \leq \xi$.

We denote this disjoint ensemble regressor as *partitioned certified regression* (PCR). Suppl. Sec. E proves that when regression is used for binary classification, PCR and DPA yield identical robustness guarantees (R).

B. Weighted Partitioned Certified Regression

Levine and Feizi only consider the maximally pessimistic unit-cost assumption. For a feature vector x_{te}, it may take multiple training set insertions/deletions to corrupt a submodel's prediction. For example, Theorems 7 and 8 prove that IBL predictions are robust to multiple training set modifications.

Fixing the regressor's overall architecture, one obvious way to improve certified robustness R is to improve the robustness certifier. Below, we introduce tighter analysis of each PCR submodel's pointwise robustness so as to move beyond unit cost. Let $r_t \in \mathbb{N}$ denote the minimum number of insertions/deletions required to change[4] the submodel enough where $f_t(x_{\text{te}}) > \xi$. By definition, if $f_t(x_{\text{te}}) > \xi$ without any training set modifications, $r_t = 0$. When $\exists_t\, r_t > 1$, better certified guarantees are possible through a weighted framework. Theorem 10 directly applies Lemma 5's weighted swap paradigm to adapt PCR (and DPA) to weighted perturbation costs. We denote this extension *weighted partitioned certified regression* (W-PCR).

Theorem 10. *For $x_{te} \in \mathcal{X}$, $\xi \in \mathbb{R}$, and function h_{tr} that partitions set $\mathcal{S}$ into disjoint blocks $S^{(1)}, \ldots, S^{(T)}$, let f be an ensemble of T submodels where T is odd. Each deterministic submodel f_t is trained on block $S^{(t)}$ and requires at least $r_t \in \mathbb{Z}_+$ modifications to $S^{(t)}$ for $f_t(x_{te}) > \xi$. For $\mathcal{R} \coloneqq \{r_t : f_t(x_{te}) \leq \xi\}$, let $\widetilde{\mathcal{R}}_1$ be $\mathcal{R}$'s smallest $|\mathcal{R}| - \lceil \frac{T}{2} \rceil + 1$ values. Given $f(x_{te}) \coloneqq \text{med}\{f_1(x_{te}), \ldots, f_T(x_{te})\} \leq \xi$, if model f is trained on a modified $\mathcal{S}$ where the total number of inserted and deleted training instances does not exceed*

$$R = \sum_{r \in \widetilde{\mathcal{R}}_1} r - 1, \tag{9}$$

it is guaranteed that $f(x_{te}) \leq \xi$.

It can be easily shown that W-PCR always yields certified robustness at least as good as PCR. Although proposed in the context of regression, our weighted formulation also notably improves certified classification as shown in Sec. X-B.

C. Computational Complexity

Both PCR and W-PCR require training $\mathcal{O}(T)$ models. As established by Lemmas 3 and 5, the computational complexity of PCR and W-PCR (resp.) to certify each ensemble prediction is $\mathcal{O}(T)$ [27] – the same complexity as DPA.[5]

VIII. Certified Regression Using Overlapping Training Data

This section reduces certified regression to a third certified classifier, specifically Wang et al.'s [26] reformulation of DPA where the submodels are trained on overlapping data. This makes the submodels interdependent, meaning one training set modification may alter multiple submodel predictions. Fig. 5 visualizes an ensemble trained on overlapping training sets. Again, T is the number of submodels.[6] Function $h_{\text{tr}} : \mathcal{Z} \to [m]$ still partitions the instance space into m disjoint blocks, where $m \geq T$. Following Wang et al., a second deterministic function $h_f : [m] \to 2^{[T]}$ maps each training set block to one or more submodel training sets. Formally, submodel

[4] Certified robustness R is the total number of training set modifications that can be made with it remaining guaranteed that $f(x_{\text{te}}) \leq \xi$. In contrast, r_t is minimum the number of modifications needed to *perturb* submodel t's prediction enough that $f_t(x_{\text{te}}) > \xi$. If R_t were the certified robustness of just *submodel* t, then $r_t = R_t + 1$. r_t's definition here follows related work [47].

[5] Not included in W-PCR's complexity is the time to determine $r_1, \ldots, r_T$.

[6] In practice, for overlapping certified regression to guarantee better robustness than (W-)PCR the number of submodels generally must increase by several folds over partitioned regression.

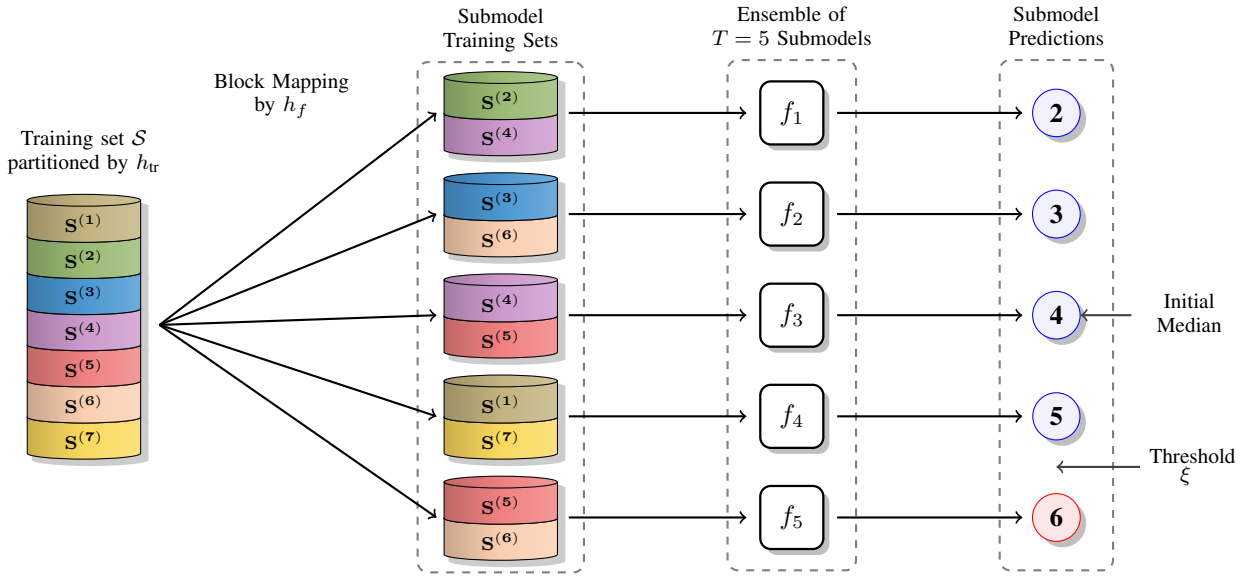


Fig. 5: **Overlapping Certified Ensemble**: Simple visualization of the ensemble architecture for (weighted) overlapping certified regression. Function h_{tr} partitions training set $\mathcal{S}$ into ($m = 7$) blocks. Function h_f defines each of the $T = 5$ submodel training sets, $\mathcal{D}_1, \ldots, \mathcal{D}_5$. The ensemble prediction is the median submodel prediction, i.e., $f(x_{\text{te}}) \coloneqq \text{med}\,\{f_1(x_{\text{te}}), \ldots, f_T(x_{\text{te}})\}$.

f_t's training set is $\mathcal{D}_t \coloneqq \bigsqcup_{t \in h_f(j)} S^{(j)}$. Let $d^{(j)} \coloneqq |h_f(j)|$ denote $S^{(j)}$'s *spread degree*, i.e., the number of models that use $S^{(j)}$ during training. Denote the *maximum spread degree* as $d_{\max} \coloneqq \max\{d^{(1)}, \ldots, d^{(m)}\}$. The ensemble's decision function is still the median submodel prediction.

Below, we first consider certified regression on overlapping data under the unit-cost assumption. We then improve overlapping regression by leveraging our weighted reformulation.

A. Overlapping Certified Regression

Irrespective of whether the submodels are trained on disjoint or overlapping data, under the unit-cost assumption, at least $|\mathcal{V}_l| - \lceil \frac{T}{2} \rceil$ submodel predictions must exceed ξ to perturb the ensemble's median. Observe that each submodel training set $\mathcal{D}_t \subset \mathcal{S}$ is composed of one or more dataset blocks. Perturbing *any* block in $\mathcal{D}_t$ is sufficient to perturb the submodel's prediction, with an optimal attacker *minimizing the number of training set (block) modifications*.

If the goal were to perturb all T submodels, then for arbitrary block mapping function h_f, determining the minimum number of blocks that need to be modified reduces to *minimum set cover*, which is NP-hard [48]. Specifically, the set to cover is $\mathcal{T}_l \coloneqq \{t : f_t(x_{\text{te}}) \leq \xi\}$, i.e., the submodels predicting at most ξ, and the collection of subsets is $\mathbf{S} \coloneqq \big\{ \{j : t \in h_f(j)\} : f_t(x_{\text{te}}) \leq \xi \big\}$, which contains the dataset blocks each relevant submodel is trained on.

However, recall that for median perturbation under unweighted swaps, we only need to perturb (i.e., cover) $|\mathcal{V}_l| - \lceil \frac{T}{2} \rceil$ submodels – not all of them. Therefore, rather than mapping to set cover, our problem reduces to the related problem of *partial set cover*, where only a constant fraction of the instances (i.e., submodels) need to be covered. For arbitrary block mapping function h_f, Lemma 11 below establishes that finding the optimal R here is NP-hard [49, 50].

Lemma 11. *Finding optimal certified robustness R for overlapping certified regression is NP-hard.*

Although our problem is NP-hard, it is polynomial-time approximable. Specifically, the approximation uses the famous greedy set-cover algorithm where in each iteration, the subset (training block $S^{(j)}$) covering the most remaining elements (submodels) is selected [49, 51]. Let G denote the bound found by this greedy method, and define $\Delta \coloneqq |\mathcal{V}_l| - \lceil \frac{T}{2} \rceil$. Then for the non-naive case where $\Delta \geq 2$,

$$R \geq \left\lceil \frac{G}{\min\{H(d_{\max}), \ln \Delta - \ln \ln \Delta + 3 + \ln \ln 32 - \ln 32\}} \right\rceil, \tag{10}$$

where $H(d_{\max})$ is the $d_{\max}$-th harmonic number. This bound follows directly from partial set cover *approximation factor* analysis [48, Thm. 4; 49, Thm. 3].[7] Slavík [48] shows that the difference between this approximation factor's overall lower and upper bound is only roughly 1.1, meaning this general approximation is quite good overall.

However, in most cases, the performance advantage of overlapping versus disjoint unit-cost regressors is small enough that the greedy optimality gap wipes out all gains. Instead, we rely on Fig. 6's integer linear program (ILP) to bound R in the overlapping case.[8] This ILP is directly adapted from standard partial set-cover, where for unit costs $\forall_t\, r_t = 1$.

While the ILP is still NP-hard in the worst case, modern LP solvers often find a (near) optimal solution in reasonable time

[7] Eq. (10)'s bound is tighter (often significantly so) than the much more famous approximation factor, $H(\Delta)$, of Johnson [52] and Lovász [53].

[8] Fig. 6 jointly formulates calculating R under unit and weighted costs.

(e.g., a few seconds) [54]. In cases where finding true robustness R is computationally expensive, these solvers generally return guaranteed bounds on R that are (much) better than the greedy approximation [55].[9] We refer to this unit-cost, ILP-based approach as *overlapping certified regression* (OCR).

B. Weighted Overlapping Certified Regression

Recall that Sec. VII-B improves certified regressor PCR by reformulating DPA so as not to be restricted by the unit-cost assumption. Here, we follow the same approach of improving certified regressor OCR by generalizing Wang et al.'s [26] certified classifier to non-unit costs.

As with W-PCR earlier, $r_t > 1$ entails that submodel f_t's training set must be modified at least r_t times for $f_t(x_{\text{te}}) > \xi$. This prevents weighted overlapping regression from applying partial set cover since each submodel f_t now has a *coverage* requirement. Instead, *partial set multicover* (PSMC) generalizes partial set cover to support coverage requirements $r_t \geq 0$ [47, 56], and we adapt PSMC to weighted, overlapping regression. PSMC, and by extension our task, is provably hard.

Corollary 12. *Finding the optimal certified robustness R for weighted overlapping certified regression is NP-hard.*

PSMC is far less studied than (partial) set cover. PSMC is polynomial-time approximable – albeit with worse known bounds than partial set cover. Ran et al. [47] provide the best-known PSMC bounds; their method is much more complicated than greedy partial set cover and relies on a reduction to another NP-hard problem, minimum densest subcollection. Let G be the solution generated by Ran et al.'s algorithm, then

$$R \geq \left\lceil \frac{G}{4 \lg T H(d_{\max}) \ln \Delta + 2 \lg T \sqrt{T}} \right\rceil. \tag{11}$$

Like with unweighted overlapping regression, Eq. (11)'s approximation factor is large enough that it usually wipes out the performance gains derived from weighted costs. Instead, we use Fig. 6's ILP to bound R in accordance with Lem. 5. In the ILP, $\sigma = 1$ in the weighted case and 0 otherwise. Hence, at least $(|\mathcal{V}_{\text{l}}| - \lceil \frac{T}{2} \rceil + 1)$ submodels must be covered (i.e., perturbed) in the weighted case. Following Eq. (5), sum $\sum_{j=1}^{m} \omega^{(j)}$ is decremented by one in the ILP.

We refer to this overlapping ILP-based approach as *weighted overlapping certified regression* (W-OCR).

C. Computational Cost

See suppl. Sec. I-E for an empirical evaluation and extended discussion of the OCR & W-OCR ILP execution time.

IX. Certifying Any Model Beyond Unit Cost

The preceding sections describe the benefits of having more robust ensemble components (i.e., $r_t > 1$) but do not address how to find r_t. Apart from IBLs and ensembles, the two methods we focus on in this work, we know of no general method for computing insertion/deletion robustness efficiently. We attribute this scarcity to the task's difficulty. Nonetheless,

[9] Sec. X's experiments use a fixed time limit to ensure tractability.

$$\begin{aligned}
\min \quad & R = \sum_{j=1}^{m} \omega^{(j)} - \sigma && && \text{(12a)}\\
\text{s.t.} \quad & \mathcal{T}_{\text{l}} = \{t : f_t(x_{\text{te}}) \leq \xi\}, && && \text{(12b)}\\
& r_{\max} = \max\{r_t : t \in [T]\} && && \text{(12c)}\\
& \sigma = \mathbb{1}[r_{\max} > 1] && && \text{(12d)}\\
& \sum_{t \in \mathcal{T}_{\text{l}}} \delta_t \geq |\mathcal{V}_{\text{l}}| - \left\lceil \frac{T}{2} \right\rceil + \sigma, && \text{(Median perturb.)} && \text{(12e)}\\
& r_t \delta_t \leq \sum_{S^{(j)} \subseteq \mathcal{D}_t} \omega^{(j)}, && t \in \mathcal{T}_{\text{l}} && \text{(12f)}\\
& \delta_t \in \{0,1\}, && t \in \mathcal{T}_{\text{l}} && \text{(12g)}\\
& \omega^{(j)} \in \{0, \ldots, r_{\max}\}, && j \in [m] && \text{(12h)}
\end{aligned}$$

Fig. 6: **Overlapping Certified Regression Integer Linear Program**: Adapted from the partial set (multi)cover integer linear program. Calculates certified robustness R for both OCR and W-OCR with indicator variable σ adjusting the program to account for weighted costs. For arbitrary feature vector x_{te}, $\mathcal{T}_{\text{l}}$ is the set of submodels that predict $f_t(x_{\text{te}}) \leq \xi$. Variable $\omega^{(j)}$ contains the number of modifications made to training set block $S^{(j)}$. Binary variable $\delta_t = 1$ if submodel f_t has been sufficiently modified for $f_t(x_{\text{te}}) > \xi$ and 0 otherwise.

we believe this work shows that certification beyond unit cost merits future study. This section explores certifying beyond unit cost from two perspectives. First, we consider the obvious idea of combining IBLs with ensembles and explain why that performs poorly. Next, we propose a simple, general approach to certify any (sub)model beyond unit cost, albeit with a (slightly) more restricted threat model.

A. Combining Instance-Based Learners & Ensembles

The points raised below apply to both fixed-population and region-based IBLs. We exclusively discuss kNN-CR here with the extension to other certified IBLs straightforward.

In practice, function h_{tr} partitions instance space $\mathcal{Z}$ uniformly at random (u.a.r.) into m approximately equal-sized regions. For simplicity and w.l.o.g., consider an ensemble of kNN-CR submodels trained on disjoint subsets where $T = m$.

Let k' and R denote the neighborhood size and certified robustness (resp.) of a kNN-CR model trained on i.i.d. training set $\mathcal{S}$. If $\mathcal{S}$ is partitioned u.a.r. to train T kNN-CR submodels each with $k \approx \frac{k'}{T}$, then each submodel's expected robustness is roughly $\frac{R}{T}$. In the best case for the defender ($\forall_t\ f_t(x_{\text{te}}) \leq \xi$), an adversary only needs to perturb at most $\lceil \frac{T}{2} \rceil$ submodels. Combining the above with Theorem 10 for W-PCR, this kNN-CR ensemble's expected certified robustness is approximately

$$\left\lceil \frac{T}{2} \right\rceil \left(\frac{R}{T} \right) - 1 = \frac{R}{2} + \frac{R}{2T} - 1 < R. \tag{13}$$

As $n, T \to \infty$, then by Eq. (13), a kNN-CR ensemble's expected robustness *decreases by 50% versus the single kNN-CR model baseline*. Intuitively, for ensembles, an adversary only needs to directly attack about half of the submodels

and by extension half of the training data. In contrast, when there is only a single kNN-CR model trained on all of $\mathcal{S}$, the adversary must attack the whole training set.

B. Certifying Non-Unit Costs by Construction

Since IBLs are a poor candidate to marry with ensembles, we need an alternative approach to certify a model's robustness beyond $r = 1$. Given the dearth of existing methods (known to us), we fill in the gap and propose a simple, general-purpose method to *certify robustness against arbitrary deletions*.

To be clear, this is a (slightly) restricted version of the full threat model considered so far, which allows arbitrary insertions and deletions. Nonetheless, this restricted threat model still has broad applicability. For example, an adversary may only be able to insert poisoned instances into a training set but not delete clean ones [2, 3, 5, 12, 19].

Motivated by Cook and Weisberg's [57] classic *case deletion diagnostics*, we use a constructive proof to certify a (sub)model's pointwise robustness under instance deletions. Consider training $(n+1)$ deterministic models – one model using full set $\mathcal{S} = \{(x_i, y_i)\}_{i=1}^n$ and another n models on each of the leave-one-out subsets $\mathcal{S} \setminus (x_i, y_i)$ for all $i \in [n]$. If all $(n+1)$ trained models make the same prediction (e.g., a value not exceeding ξ for some x_{te}), then by construction, the model trained on all of $\mathcal{S}$ has, at minimum, $r = 2$ for arbitrary deletions. Lemma 13 generalizes the above for an arbitrary number of deletions $r < n$.

Lemma 13. *For $x_{te} \in \mathcal{X}$, training set $\mathcal{S}$ where $2^{\mathcal{S}}$ is its power set, $r \in [|\mathcal{S}| - 1]$, and $\xi \in \mathbb{R}$, denote a deterministic model trained on subset $S \subseteq \mathcal{S}$ as f_S. Given $\forall_{S' \in 2^{\mathcal{S}}} |S'| < r \implies f_{\mathcal{S} \setminus S'}(x_{te}) \leq \xi$, then for any $\widetilde{\mathcal{S}} \subset \mathcal{S}$, if $f_{\widetilde{\mathcal{S}}}(x_{te}) > \xi$ then at least r instances from $\mathcal{S}$ were deleted in $\widetilde{\mathcal{S}}$.*

A strength of Lemma 13 is its flexibility; it can be adapted to any model class, including both classifiers and regressors. Its clear limitation is its computational complexity.

Computational Complexity: Certifying $r > 1$ requires training $\mathcal{O}(n^{(r-1)})$ models; this is a one-time, amortized cost.

Consider separately the cost to certify each prediction. During inference, the $\mathcal{O}(n^{(r-1)})$ models are checked. While this may be problematic in some cases, it should be contextualized. Recall that Sec. VI explores IBLs like kNN, which have inference complexity $\mathcal{O}(n)$. Therefore, our method to certify $r = 2$ has the same time complexity *during inference* as kNN.

C. More Submodels vs. Weighted Costs

Increasing submodel count T and using weighted costs are partially conflicting approaches to increase R. A natural question is which of the two approaches yields better certified robustness. Above, we explain why increasing T is a poor strategy for IBLs. For ensembles, increasing T generally means that each submodel is trained on fewer data.

As an intuition, consider when $\forall_t r_t = 2$. For a unit-cost ensemble to certify equivalent R, submodel count T must about double, and each submodel is trained on 50% fewer data, which can significantly degrade submodel performance. In contrast, weighted costs with $r = 2$ reduces submodel training set sizes by 1 (Lem. 13). By training weighted submodels on much more data, weighted submodels can outperform submodels from ensembles with larger T. This improved submodel performance can in turn improve certified robustness.

X. Evaluation

This section evaluates our five primary certified regressors: kNN-m certified regression (kNN-CR), partitioned certified regressors PCR & W-PCR as well as overlapping certified regressors OCR & W-OCR. Additional experimental results are in the supplement, including full kNN-CR certification plots (I-B), alternate ξ value analysis (I-C), model training times (I-D), ILP execution time (I-D), etc.

To the extent of our knowledge, we propose the first pointwise certified regression methods that make no assumptions about the test distribution or model architecture. Without a clear baseline, we compare our five methods against each other. As a reference on the clean-data performance, we report each dataset's "*uncertified*" (non-robust) accuracy.

A. Experimental Setup

Due to space, most evaluation setup details (e.g., hyperparameters) are deferred to suppl. Sec. H with a brief summary below. For each experiment in this section, at least ten trials were performed. To improve readability, we only report the mean values below with variances in suppl. Sec. I-A.

Dataset Configuration Each (sub)model is trained on $\frac{1}{q}$-th of the training data, where $q \in \mathbb{Z}_{>0}$. For kNN-CR, always $q = 1$. For our four ensemble-based methods (W-)PCR and (W-)OCR, q can significantly affect the ensemble's accuracy and best-case certified robustness (R). As such, for each dataset, we report results with three different q values. For all ensembles, function h_{tr} partitions training set $\mathcal{S}$ u.a.r.[10]

For our partitioned regressors (W-)PCR, $\mathcal{S}$ is split into q blocks, with $T = q$. For our overlapping regressors (W-)OCR, we followed Wang et al.'s [26] overlapping certified classifier evaluation. Specifically, $\mathcal{S}$ is partitioned into qd blocks u.a.r. All blocks have fixed spread degree $d > 1$ (see Tab. I), and h_f assigns blocks to submodels at random. Hence, each overlapping ensemble necessarily has $T = qd$ submodels.

Submodel Architectures To demonstrate their generality, our ensemble methods use two different submodel architectures, namely ridge regression and XGBoost [21] gradient-boosted forests. Model determinism is enforced via a fixed random seed. Below, we report whichever submodel architecture performed the best on a held-out validation set.

Evaluation Metric For each test instance $(x_{\text{te}}, y_{\text{te}})$, our goal is to determine the largest pointwise certified robustness R that guarantees $\xi_{\text{l}} \leq f(x_{\text{te}}) \leq \xi_{\text{u}}$. Throughout this evaluation, $\xi_{\text{l}} := y_{\text{te}} - \xi$ and $\xi_{\text{u}} := y_{\text{te}} + \xi$. These bounds are w.r.t. each test example's *true target value* y_{te}, not predicted value $f(x_{\text{te}})$.

[10] Each dataset's largest q value maximized the ensembles' certified robustness (R). For each dataset, we also report small and medium q values. In practice, q should be as small as possible while guaranteeing sufficient robustness given each application's maximum anticipated poisoning rate.

TABLE I: **Evaluation Dataset Summary**: Training set size (n), data dimension, overlapping spread degree (d), error threshold (ξ), and submodel architecture for the six datasets. Error thresholds that are a percentage of each instance's true target value are denoted $X\% \cdot y$. Alternate ξ values are evaluated in suppl. Sec. I-C.

Dataset	Size (n)	Dim.	Deg. (d)	Error (ξ)	Submodel
Ames	2.6k	253	17	$15\% \cdot y$	XGBoost
Austin	12k	315	13	$15\% \cdot y$	XGBoost
Diamonds	48k	26	9	$15\% \cdot y$	Ridge
Weather	308k	140	5	3° C	Ridge
Life	2.6k	204	13	3 years	XGBoost
Spambase	4.1k	57	17	0	Ridge

Therefore, a large certified robustness R means that the *prediction is both accurate and stable*. Here, *error threshold* ξ may be a specific fraction (e.g., 15%) of each instance's target value y_{te} or a fixed value for the entire dataset (see Table I). In practice, the appropriate ξ value is application specific.

Our evaluation metric is *certified accuracy*, which is the fraction of instances with robustness $R \geq \psi$ for $\psi \in \mathbb{N}$. In each trial, we calculated the certified robustness (R) for at least 100 random test instances and report the mean certified accuracy across all trials. See suppl. Sec. I-A for the certified accuracy variance. Note that existing certified classifiers were previously evaluated using certified accuracy [24, 25, 26] with $\xi = 0$, i.e., the predicted label must match true label y_{te}.

Datasets Our certified regressors are evaluated on six datasets: five regression and one binary classification. Like previous work [39], the datasets are preprocessed where all categorical features are transformed into one-hot encodings. Table I summarizes each dataset's key attributes, including its size, error threshold (ξ), ensemble submodel architecture, etc. A brief description of each dataset is below.

Ames [58] and Austin [59] estimate home prices in two American cities. Diamonds [60] predicts a diamond's price based on attributes such as cut, color, carat, etc. Weather [61] estimates ground temperature (in degrees Celsius) using date, time, and longitude/latitude information. For computational efficiency, Weather's size was downsampled by $10\times$ u.a.r. Life [62] estimates life expectancy (in years) using epidemiological and other national statistics. Spambase [63] is a binary classification dataset where emails are labeled as either spam or ham. Spambase's positive training prior is 39%.

Certifying $r > 1$ For our two weighted methods, W-PCR and W-OCR, our evaluation attempts to certify each submodel's robustness against deletions up to $r = 2$.

B. Analyzing the Certified Accuracy

Figure 7 visualizes our methods' mean certified accuracy for the six datasets. Due to space, the corresponding numerical values, including variance, appear in Sec. I-A. Below, we briefly summarize the experiments' primary takeaways.

Takeaway #1: *Both our ensemble and IBL regressors certify non-trivial fractions of the training set.* For the Ames and Life datasets, W-OCR certifies 50% of test predictions up to 1% training set corruption. Similarly, kNN-CR certifies 30% of predictions on Ames up to 4% corruption. These certified guarantees are without explicit assumptions about the data distribution or, in the case of ensembles, the submodel's architecture. For other datasets, we certify predictions up to hundreds or thousands of training set modifications.

Takeaway #2: *Ensemble regressors have better peak performance.* Across all six datasets, the ensemble-based methods all had better peak certified accuracy than kNN-CR. The performance gap was as large as $3.5\times$ and is not primarily due to feature dimension as kNN-CR performed worst on Diamonds, which has the smallest dimension by far.

Takeaway #3: W-OCR *achieves the largest certified robustness (R) amongst the ensemble methods.* This is observed using each dataset's largest q value. For all six datasets, there is a (significant) gap between W-OCR (—) and our second-best ensemble method, W-PCR (—).

Takeaway #4: kNN-CR *achieves the largest certified robustness.* Although kNN-CR certifies (far) fewer instances than the ensembles, for instances that it can certify, its maximum robustness R is generally far larger than that of W-OCR. For example with Weather, kNN-CR's maximum R is $5\times$ larger than W-OCR's. Suppl. Sec. I-B best visualizes this trend in its plots of kNN-CR's full certified accuracy.

Takeaway #5: W-OCR *achieves state-of-the-art certified accuracy for binary classification.* While regression is this work's primary focus, recall that binary classification can be solved by a regressor. For binary classification, kNN-CR's R is identical to Jia et al.'s [24] kNN classifier; PCR certifies equivalent robustness as DPA, and OCR very closely approximates Wang et al.'s [26] overlapping method. Observe that W-OCR outperforms the unweighted ensembles and kNN-CR on Spambase's [63] two largest q values. Note that Spambase's maximum q value cannot be increased further without severely degrading submodel performance. This provides empirical evidence for Sec. IX-C's claim that a weighted strategy can outperform increasing submodel count T.

Takeaway #6: q *can significantly affect certified accuracy.* Previous certified classifier evaluations [24, 25, 26] underexplore q's role. Those works primarily evaluate vision datasets where the training data is supplemented by (1) using a pre-trained model to extract much better features [24, 41] or (2) using vision data augmentation [25, 26]. For the tabular datasets evaluated here, such options are not as available.

Without such augmentation, increasing q can significantly degrade an ensemble's peak certified accuracy. As an example, the ensembles' peak certified accuracy can decline by up to 28% between training a model on all of $\mathcal{S}$ versus a dataset's maximum q value (compare to uncertified accuracy ····· in Fig. 7). Therefore, when thinking about certified classifiers and regressors, always consider the potential benefits of external (clean) data augmentation. For instance, in our experiments, XGBoost certified ensembles' accuracy improved by multiple percent when using *mixup* data augmentation [64].

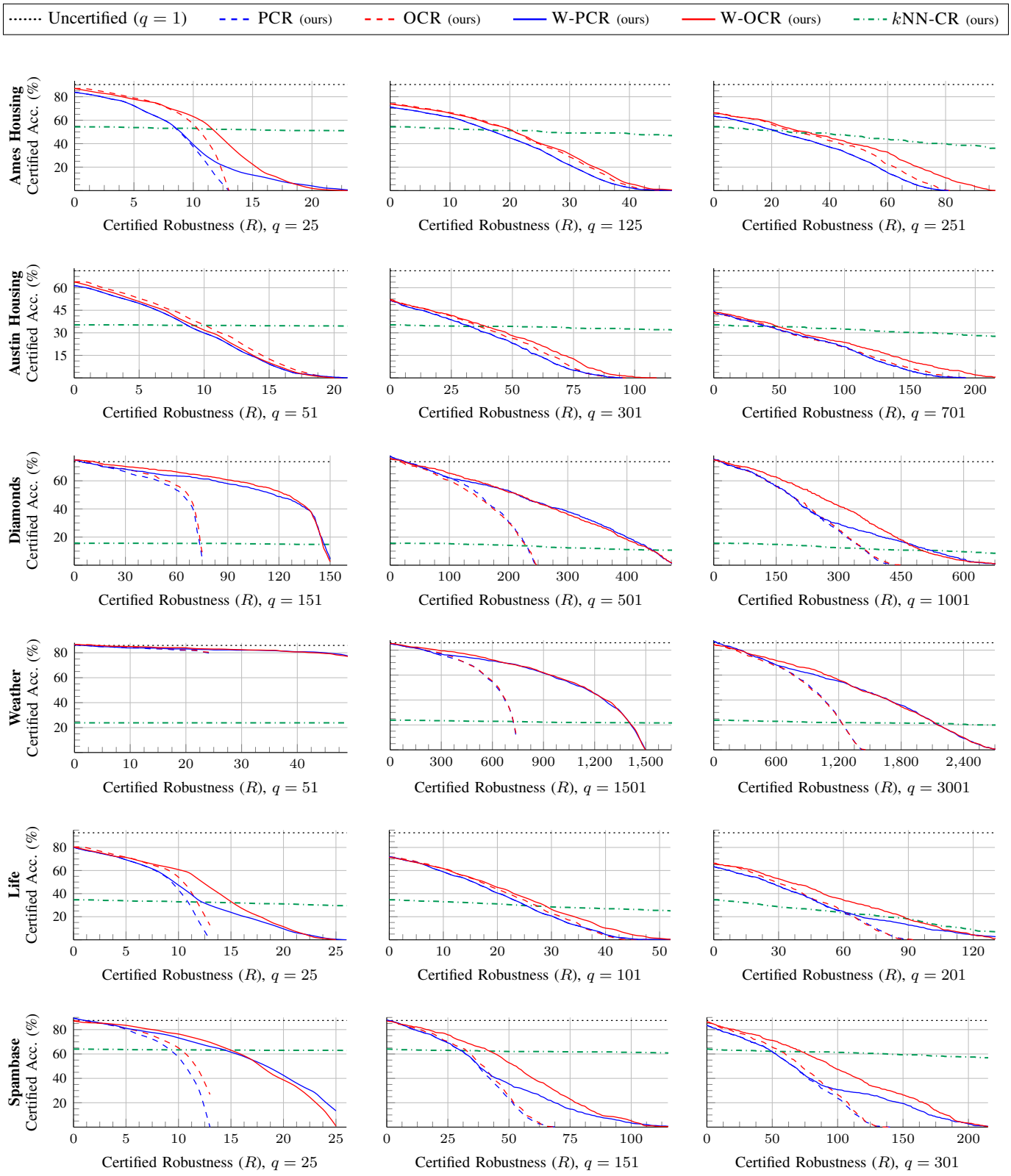


Fig. 7: **Certified Accuracy**: Mean certified accuracy (larger is better) for our five primary certified regressors. kNN-CR is always trained on all of training set $\mathcal{S}$ (i.e., $q = 1$). Ensemble submodels are trained on $\frac{1}{q}$-th of $\mathcal{S}$, with three q values tested per dataset. The x-axis is clipped to enhance readability; see suppl. Sec. I-B for kNN-CR's full results. The best performing method depends on the target certified robustness R. For smaller R values, W-OCR achieves the best certified accuracy. For larger R values, kNN-CR outperforms the ensemble methods. This result aligns with previous findings on certified classification [24]. Sec. X-B summarizes these experiments' primary takeaways. See Sec. I-A for the numerical results, including variance.

XI. Conclusions

This paper describes a novel reduction from certified regression to certified classification based on median perturbation. Applying this reduction, we propose six new certified regressors that require no assumptions about the data distribution or model architecture. As improved voting-based, certified classifiers are proposed in the future, our reduction can be applied to those methods too. This enables certified regression to keep pace with the rapid advances in certified classification.

While this work focuses on certified defenses against poisoning attacks, some certified *evasion* defenses also rely on voting-based guarantees [24, 42]. Our reduction from certified regression to certified classification applies to those certified evasion defenses as well.

Lastly, our empirical results show that improved certified guarantees are possible when the unit-cost assumption is replaced by our tighter weighted analysis. These certification gains apply to both classification and regression, but Sec. IX-B's approach is computationally expensive. We advocate for better methods that efficiently certify beyond $r = 1$.

Acknowledgments

The authors thank Jonathan Brophy and Will Bolden for helpful discussions and feedback on earlier drafts of this manuscript.

This work was supported by a grant from the Air Force Research Laboratory and the Defense Advanced Research Projects Agency (DARPA) — agreement number FA8750-16-C-0166, subcontract K001892-00-S05, as well as a second grant from DARPA, agreement number HR00112090135. This work benefited from access to the University of Oregon high performance computer, Talapas.

References

[1] B. Biggio, B. Nelson, and P. Laskov, "Poisoning attacks against support vector machines," in *Proceedings of the 29th International Conference on Machine Learning*, ser. ICML'12. Edinburgh, Great Britain: PMLR, 2012.

[2] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, "Targeted backdoor attacks on deep learning systems using data poisoning," 2017, arXiv.

[3] A. Shafahi, W. R. Huang, M. Najibi, O. Suciu, C. Studer, T. Dumitras, and T. Goldstein, "Poison frogs! Targeted clean-label poisoning attacks on neural networks," in *Proceedings of the 32nd Conference on Neural Information Processing Systems*, ser. NeurIPS'18. Montreal, Canada: Curran Associates, Inc., 2018.

[4] W. R. Huang, J. Geiping, L. Fowl, G. Taylor, and T. Goldstein, "MetaPoison: Practical general-purpose clean-label data poisoning," in *Proceedings of the 34th Conference on Neural Information Processing Systems*, ser. NeurIPS'20. Virtual Only: Curran Associates, Inc., 2020.

[5] E. Wallace, T. Z. Zhao, S. Feng, and S. Singh, "Concealed data poisoning attacks on NLP models," in *Proceedings of the North American Chapter of the Association for Computational Linguistics*, ser. NAACL'21, 2021.

[6] R. S. S. Kumar, M. Nyström, J. Lambert, A. Marshall, M. Goertzel, A. Comissoneru, M. Swann, and S. Xia, "Adversarial machine learning – industry perspectives," in *Proceedings of the 2020 IEEE Security and Privacy Workshops*, ser. SPW'20, 2020.

[7] P. Lee, "Learning from Tay's introduction," 2016. [Online]. Available: https://blogs.microsoft.com/blog/2016/03/25/learning-tays-introduction/

[8] N. Carlini, "On evaluating adversarial robustness," 2019. [Online]. Available: https://youtu.be/-p2il-V-0fk?t=1574

[9] B. Tran, J. Li, and A. Madry, "Spectral signatures in backdoor attacks," in *Proceedings of the 32nd Conference on Neural Information Processing Systems*, ser. NeurIPS'18. Montreal, Canada: Curran Associates, Inc., 2018.

[10] N. Peri, N. Gupta, W. R. Huang, L. Fowl, C. Zhu, S. Feizi, T. Goldstein, and J. P. Dickerson, "Deep k-NN defense against clean-label data poisoning attacks," in *Proceedings of the ECCV Workshop on Adversarial Robustness in the Real World*, ser. AROW'20, Virtual Only, 2020.

[11] L. Zhu, R. Ning, C. Xin, C. Wang, and H. Wu, "CLEAR: Clean-up sample-targeted backdoor in neural networks," in *Proceedings of the 18th International Conference on Computer Vision*, ser. ICCV'21, 2021.

[12] Z. Hammoudeh and D. Lowd, "Identifying a training-set attack's target using renormalized influence estimation," in *Proceedings of the 29th ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS'22. Association for Computing Machinery, 2022.

[13] Y. Gao, C. Xu, D. Wang, S. Chen, D. C. Ranasinghe, and S. Nepal, "STRIP: A defence against Trojan attacks on deep neural networks," in *Proceedings of the 35th Annual Computer Security Applications Conference*, ser. ACSAC'19. San Juan, Puerto Rico, USA: Association for Computing Machinery, 2019.

[14] K. A. Lai, A. B. Rao, and S. Vempala, "Agnostic estimation of mean and covariance," in *Proceedings of the 57th Annual Symposium on Foundations of Computer Science*, ser. FOCS'16, 2016, pp. 665–674.

[15] J. Steinhardt, P. W. Koh, and P. Liang, "Certified defenses for data poisoning attacks," in *Proceedings of the 31st Conference on Neural Information Processing Systems*, ser. NeurIPS'17. Long Beach, California, USA: Curran Associates, Inc., 2017.

[16] B. Wang, X. Cao, J. Jai, and N. Z. Gong, "On certifying robustness against backdoor attacks via randomized smoothing," in *Proceedings of the CVPR Workshop on Adversarial Machine Learning in Computer Vision*, 2020.

[17] M. Weber, X. Xu, B. Karlaš, C. Zhang, and B. Li, "RAB: Provable robustness against backdoor attacks," in *Proceedings of the 44th IEEE Symposium on Security and Privacy*, ser. SP'23. IEEE, 2023.

[18] L. Liu, Y. Shen, T. Li, and C. Caramanis, "High dimensional robust sparse regression," in *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, ser. AISTATS'20, 2020.

[19] C. Liu, B. Li, Y. Vorobeychik, and A. Oprea, "Robust linear regression against training data poisoning," in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, ser. AISec'17. New York, NY, USA: Association for Computing Machinery, 2017.

[20] M. Jagielski, A. Oprea, B. Biggio, C. Liu, C. Nita-Rotaru, and B. Li, "Manipulating machine learning: Poisoning attacks and countermeasures for regression learning," in *Proceedings of the 2018 IEEE Symposium on Security and Privacy*, ser. SP'18, 2018.

[21] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD'16. New York, NY, USA: Association for Computing Machinery, 2016.

[22] L. O. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "CatBoost: Unbiased boosting with categorical features," in *Proceedings of the 32nd Conference on Neural Information Processing Systems*, ser. NeurIPS'18. Montreal, Canada: Curran Associates, Inc., 2018.

[23] S. Dasgupta, C. H. Papadimitriou, and U. V. Vazirani, *Algorithms*. McGraw-Hill, 2008.

[24] J. Jia, Y. Liu, X. Cao, and N. Z. Gong, "Certified robustness of nearest neighbors against data poisoning and backdoor attacks," in *Proceedings of the 36th AAAI Conference on Artificial Intelligence*, ser. AAAI'22, 2022.

[25] A. Levine and S. Feizi, "Deep partition aggregation: Provable defenses against general poisoning attacks," in *Proceedings of the 9th International Conference on Learning Representations*, ser. ICLR'21, Virtual Only, 2021.

[26] W. Wang, A. Levine, and S. Feizi, "Improved certified defenses against data poisoning with (deterministic) finite aggregation," in *Proceedings of the 39th International Conference on Machine Learning*, ser. ICML'22, 2022.

[27] M. Blum, R. W. Floyd, V. Pratt, R. L. Rivest, and R. E. Tarjan, "Time bounds for selection," *Journal of Computer and System Sciences*, vol. 7, no. 4, p. 448–461, 1973.

[28] P. J. Huber, "Robust estimation of a location parameter," *Annals of Mathematical Statistics*, vol. 35, no. 1, pp. 73–101, 1964.

[29] A. E. Beaton and J. W. Tukey, "The fitting of power series, meaning polynomials, illustrated on band-spectroscopic data," *Technometrics*, vol. 16, no. 2, pp. 147–185, 1974.

[30] J. E. D. Jr. and R. E. Welsch, "Techniques for nonlinear least squares and robust regression," *Communications in Statistics - Simulation and Computation*, vol. 7, no. 4, pp. 345–359, 1978.

[31] Y. G. Leclerc, "Constructing simple stable descriptions for image partitioning," *International Journal of Computer Vision*, vol. 3, no. 1, pp. 73–102, 1989.

[32] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[33] P. H. S. Torr and A. Zisserman, "MLESAC: A new robust estimator with application to estimating image geometry." *Computer Vision and Image Understanding*, vol. 78, no. 1, pp. 138–156, 2000.

[34] P. J. Rousseeuw and M. Hubert, "Robust statistics for outlier detection," *WIREs Data Mining and Knowledge Discovery*, vol. 1, no. 1, pp. 73–79, 2011.

[35] Y. Li, B. Wu, Y. Jiang, Z. Li, and S. Xia, "Backdoor learning: A survey," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[36] A. R. Klivans, P. K. Kothari, and R. Meka, "Efficient algorithms for outlier-robust regression," in *Proceedings of the 31st Conference on Learning Theory*, ser. COLT'18. PMLR, 2018.

[37] Y. Chen, C. Caramanis, and S. Mannor, "Robust high dimensional sparse regression and matching pursuit," 2013, arXiv.

[38] S. Arik and T. Pfister, "TabNet: Attentive interpretable tabular learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

[39] J. Brophy, Z. Hammoudeh, and D. Lowd, "Adapting and evaluating influence-estimation methods for gradient-boosted decision trees," 2022, arXiv.

[40] E. Rosenfeld, E. Winston, P. Ravikumar, and J. Z. Kolter, "Certified robustness to label-flipping attacks via randomized smoothing," in *Proceedings of the 37th International Conference on Machine Learning*, ser. ICML'20, vol. 119. PMLR, 2020, pp. 8230–8241.

[41] J. Jia, X. Cao, and N. Z. Gong, "Intrinsic certified robustness of bagging against data poisoning attacks," in *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, ser. AAAI'21, 2021.

[42] A. Levine and S. Feizi, "Robustness certificates for sparse adversarial attacks by randomized ablation," in *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, ser. AAAI'20. AAAI Press, 2020.

[43] J. Jia, B. Wang, X. Cao, H. Liu, and N. Z. Gong, "Almost tight ℓ_0-norm certified robustness of top-k predictions against adversarial perturbations," in *Proceedings of the 10th International Conference on Learning Representations*, ser. ICLR'22, 2022. [Online]. Available: https://openreview.net/forum?id=gJLEXy3ySpu

[44] D. W. Aha, D. Kibler, and M. K. Albert, "Instance-based learning algorithms," *Machine Learning*, vol. 6, no. 1, pp. 37–66, 1991.

[45] J. L. Bentley, "A survey of techniques for fixed radius near neighbor searching," Stanford, CA, USA, Tech. Rep., 1975.

[46] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Machine Learning*, vol. 63, no. 1, pp.

3–42, 2006.

[47] Y. Ran, Z. Zhang, S. Tang, and D.-Z. Du, "Breaking the $r_{\max}$ barrier: Enhanced approximation algorithms for partial set multicover problem," *INFORMS Journal on Computing*, vol. 33, no. 2, pp. 774–784, 2021.

[48] P. Slavík, "A tight analysis of the greedy algorithm for set cover," *Journal of Algorithms*, pp. 237–254, 1997.

[49] ——, "Improved performance of the greedy algorithm for partial cover," *Information Processing Letters*, vol. 64, no. 5, pp. 251–254, 1997.

[50] T. Elomaa and J. Kujala, *Covering Analysis of the Greedy Algorithm for Partial Cover*. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 102–113.

[51] V. Chvatal, "A greedy heuristic for the set-covering problem," *Mathematics of Operations Research*, vol. 4, no. 3, pp. 233–235, 1979.

[52] D. S. Johnson, "Approximation algorithms for combinatorial problems," *Journal of Computer and System Sciences*, vol. 9, no. 3, pp. 256–278, 1974.

[53] L. Lovász, "On the ratio of optimal integral and fractional covers," *Discrete Mathematics*, vol. 13, no. 4, pp. 383–390, 1975.

[54] Gurobi Optimization, LLC, "Gurobi Optimizer Reference Manual," 2022. [Online]. Available: https://www.gurobi.com

[55] R. J. Vanderbei, *Linear Programming: Foundations and Extensions*. Springer, 2014.

[56] Y. Shi, Y. Ran, Z. Zhang, J. Willson, G. Tong, and D.-Z. Du, "Approximation algorithm for the partial set multicover problem," *Journal of Global Optimization*, vol. 75, no. 4, pp. 1133–1146, 2019.

[57] R. D. Cook and S. Weisberg, *Residuals and Influence in Regression*. New York: Chapman and Hall, 1982.

[58] D. D. Cock, "Ames, Iowa: Alternative to the Boston Housing Data as an end of semester regression project," *Journal of Statistics Education*, vol. 19, no. 3, 2011.

[59] E. Pierce, "Austin, TX house listings," 2021. [Online]. Available: https://www.kaggle.com/datasets/ericpierce/austinhousingprices

[60] H. Wickham, *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag, 2016.

[61] A. Malinin, N. Band, Y. Gal, M. Gales, A. Ganshin, G. Chesnokov, A. Noskov, A. Ploskonosov, L. Prokhorenkova, I. Provilkov, V. Raina, V. Raina, D. Roginskiy, M. Shmatova, P. Tigas, and B. Yangel, "Shifts: A dataset of real distributional shift across multiple large-scale tasks," in *Proceedings of the 35th Conference on Neural Information Processing Systems*, ser. NeurIPS'21. Curran Associates, Inc., 2021.

[62] K. Rajarshi, "Life expectancy (WHO)," 2021. [Online]. Available: https://www.kaggle.com/datasets/kumarajarshi/life-expectancy-who

[63] M. Hopkins, E. Reeber, G. Forman, and J. Suermondt, "UCI machine learning repository: Spambase dataset," 2017. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/spambase

[64] H. Zhang, M. Cissé, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *Proceedings of the 6th International Conference on Learning Representations*, ser. ICLR'18, 2018.

[65] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and Édouard Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[66] T. Head, M. Kumar, H. Nahrstaedt, G. Louppe, and I. Shcherbatyi, "scikit-optimize: Sequential model-based optimization in Python," 2021.

Reducing Certified Regression to Certified Classification for General Poisoning Attacks

Supplemental Materials

Organization of the Appendix

APPENDIX A
NOMENCLATURE REFERENCE

TABLE II: **Nomenclature Reference**: Related symbols are grouped together. For example, the first group lists the acronyms of our primary certified regressors. Note that this table continues on the next page.

kNN-CR	Our kNN-based certified regressor (Sec. VI-A)
PCR	Our partitioned certified regressor (Sec. VII-A)
W-PCR	Our weighted-cost partitioned certified regressor (Sec. VII-B)
PCR	Our overlapping certified regressor (Sec. VIII-A)
W-PCR	Our weighted-cost overlapping certified regressor (Sec. VIII-B)
R	Pointwise certified robustness – number of possible training set insertions or deletions without violating the prediction bounds – our primary goal
$[k]$	Integer set $\{1, \ldots, k\}$
$2^{[k]}$	Power set of integer set $[k]$
$\mathbb{1}[q]$	Indicator function where $\mathbb{1}[q] = 1$ if q is true and 0 otherwise
med A	Median of (multi)set A
$H(k)$	k-th harmonic number where $H(k) = \sum_{i=1}^{k} \frac{1}{i}$
x	Feature vector
$\mathcal{X}$	Feature domain where $\forall_x\, x \in \mathcal{X}$ and $\mathcal{X} \subseteq \mathbb{R}^d$
d	Feature dimension
y	Target value
$\mathcal{Y}$	Target range where $\forall_y\, y \in \mathcal{Y}$ and $\mathcal{Y} \subseteq \mathbb{R}$
$\mathcal{Z}$	Instance space where $\mathcal{Z} \coloneqq \mathcal{X} \times \mathcal{Y}$
$(x_{\text{te}}, y_{\text{te}})$	Arbitrary test instance
$\mathcal{S}$	Training set where $\mathcal{S} \subseteq \mathcal{Z}$
n	Training set size where $n \coloneqq \lvert\mathcal{S}\rvert$
m	Number of training set blocks
h_{tr}	Training set partitioning function where $h_{\text{tr}} : \mathcal{Z} \to [m]$
j	Training set block index where $j \in [m]$
$S^{(j)}$	j-th training set block where $S^{(j)} \coloneqq \{z \in \mathcal{S} : h_{\text{tr}}(z) = j\}$ and $\forall_{j' \neq j}\, S^{(j)} \cap S^{(j')} = \emptyset$
f	Robust regressor – either an ensemble or instance-based learner – where $f : \mathcal{X} \to \mathcal{Y}$
$f(x_{\text{te}})$	Regressor f's prediction for test feature vector x_{te}
ξ	One-sided upper bound for robustness certification where $f(x_{\text{te}}) \leq \xi$
ξ_{l}	Two-sided lower bound for robustness certification where $\xi_{\text{l}} \leq f(x_{\text{te}}) \leq \xi_{\text{u}}$
ξ_{u}	Two-sided upper bound for robustness certification where $\xi_{\text{l}} \leq f(x_{\text{te}}) \leq \xi_{\text{u}}$
kNN	Vanilla k-nearest neighbors
kNN-m	k-nearest neighbors with median as the decision function
rNN	Radius nearest neighbors
$\mathcal{N}(x_{\text{te}})$	Nearest-neighbors neighborhood (multi)set for test feature vector x_{te}
T	Ensemble submodel count where by definition T is selected to be odd-valued
t	Submodel index where $t \in [T]$
f_t	t-th ensemble submodel
r_t	Number of training set modifications required to violate invariant $\xi_{\text{l}} \leq f_t(x_{\text{te}}) \leq \xi_{\text{u}}$. Note that r_t is one larger than f_t's certified robustness

TABLE II: **Nomenclature Reference (Continued)**: Related symbols are grouped together.

$r_{\max}$	Maximum submodel modification requirement where $r_{\max} \coloneqq \max_t r_t$
h_f	Overlapping training set block assignment function where $h_f : [m] \to 2^{[T]}$
$\mathcal{D}_t$	Submodel f_t's training set where for overlapping regression $\mathcal{D}_t = \bigcup_{\substack{j \in [m] \\ t \in h_f(j)}} S^{(j)}$
q	Inverse of the fraction of the training set used to train each submodel, where $\forall_t \, q = \frac{n}{\lvert\mathcal{D}_t\rvert}$
$d^{(j)}$	Spread degree of training set block $S^{(j)}$ where $d^{(j)} \coloneqq \lvert h_f(S^{(j)})\rvert$
$d_{\max}$	Maximum spread degree where $d_{\max} \coloneqq \max_j d^{(j)}$
$\mathcal{T}_{\text{l}}$	Set of ensemble submodels predicting $f_t(x_{\text{te}}) \leq \xi$ where $\mathcal{T}_{\text{l}} \subseteq [T]$
$\mathcal{V}$	Real-valued (multi)set, e.g., kNN neighborhood or ensemble submodel predictions, where $T = \lvert\mathcal{V}\rvert$
$\mathcal{V}_{\text{l}}$	Lower thresholded real-valued multiset where $\mathcal{V}_{\text{l}} \coloneqq \{\nu_t \in \mathcal{V} : \nu_t \leq \xi\}$
$\mathcal{V}_{\text{u}}$	Upper thresholded real-valued multiset where $\mathcal{V}_{\text{u}} \coloneqq \{\nu_t \in \mathcal{V} : \nu_t > \xi\}$
$\mathcal{V}_{\pm 1}$	Binarized multiset where $\mathcal{V}_{\pm 1} \coloneqq \{\operatorname{sgn}(\nu_t) : \nu_t \in \mathcal{V}\}$
$\widetilde{\mathcal{V}}$	Adversarially perturbed real-valued (multi)set formed from (multi)set $\mathcal{V}$
$\mathcal{R}$	Weight set where $\mathcal{R} \coloneqq \{r_t : t \in [T]\}$
$\mathcal{R}_{\text{l}}$	Weight set corresponding to values set $\mathcal{V}_{\text{l}}$ where $\mathcal{R}_{\text{l}} \coloneqq \{r_t \in \mathcal{R} : \nu_t \in \mathcal{V}\}$
Δ	Midpoint distance where $\Delta \coloneqq \lvert\mathcal{V}_{\text{l}}\rvert - \lceil \frac{T}{2} \rceil$
$\widetilde{\mathcal{R}}_{\text{l}}$	Δ smallest values in $\mathcal{R}_{\text{l}}$
ILP	Integer linear program
$\omega^{(j)}$	ILP integral variable representing the number of instance modifications made to training set block $S^{(j)}$
δ_t	ILP binary variable which equals 1 if submodel f_t has been perturbed such that $f_t(x_{\text{te}}) > \xi$ and 0 otherwise
σ	ILP binary variable which equals 1 if in the case of weighted analysis and 0 otherwise
PSMC	Partial set multicover
G	Upper-bound on certified robustness R returned by a greedy algorithm

APPENDIX B
WORST-CASE INSERTIONS AND DELETIONS ARE INTERCHANGEABLE: PROOF

Lemma 14. *For real multiset $\mathcal{V}$ of cardinality $T > 1$, if an arbitrarily-large value is inserted into $\mathcal{V}$ or the smallest value in $\mathcal{V}$ is deleted, the resulting sets' medians are equivalent.*

Proof. For simplicity and w.l.o.g., assume $\mathcal{V}$ is ordered where $\nu_1 \leq \cdots \leq \nu_T$. Let $\alpha \in [1, T]$ denote the median's index. If T is odd, $\alpha = \lceil \frac{T}{2} \rceil$; otherwise, when T is even, $\alpha = \frac{T}{2} + \frac{1}{2}$, i.e., the midpoint between the $\frac{T}{2}$-th and $(\frac{T}{2}+1)$-th largest values in $\mathcal{V}$.

Consider first an arbitrarily-large insertion when T odd. Each insertion increases the set's cardinality by 1. When $\mathcal{V}$'s cardinality is odd, then the cardinality of this new set after the first insertion is even. Therefore, this new set's median has index

$$\alpha' \coloneqq \frac{T+1}{2} + \frac{1}{2} \qquad \triangleright \text{ Median's index for new set of even size } T+1 \tag{14}$$

$$= \left\lceil \frac{T}{2} \right\rceil + \frac{1}{2} \tag{15}$$

$$= \alpha + \frac{1}{2}. \tag{16}$$

Since $T \geq \alpha'$ and the inserted element is larger than all values in $\mathcal{V}$, the value corresponding to index $(\lceil \frac{T}{2} \rceil - \frac{1}{2})$ is equivalent for both original set $\mathcal{V}$ and the new set after the insertion.

Next, consider the deletion case for odd T. Similar to above, the cardinality of the modified set after one deletion is even; therefore, this modified set's median has index

$$\alpha'' \coloneqq \frac{T-1}{2} + \frac{1}{2} \qquad \triangleright \text{ Median's index for new set of even size } T-1 \tag{17}$$

$$= \left\lceil \frac{T}{2} \right\rceil - \frac{1}{2} \tag{18}$$

$$= \alpha - \frac{1}{2}. \tag{19}$$

This new set's cardinality is one smaller than the original set with the smallest element removed. Hence, the value corresponding to index $(\lceil \frac{T}{2} \rceil - \frac{1}{2})$ in this shrunken set equals the value at index $(\lceil \frac{T}{2} \rceil + \frac{1}{2})$ in original set $\mathcal{V}$.

Since indices α' and α'' correspond to the same value in $\mathcal{V}$, the resulting sets' medians are equivalent. □

The primary takeaway from Lemma 14 is that under the insertion/deletion paradigm, worst-case insertions and deletions are interchangeable. Note that for our purposes, there is an edge case where worst-case insertions and deletions exhibit divergent behavior. Specifically, after T deletions (i.e., all elements in $\mathcal{V}$ are removed), the median of an empty set is not generally defined. In contrast, the median after T arbitrarily-large insertions is itself arbitrarily large. For consistency, we define the empty set's median as ∞ to match the insertion case.

APPENDIX C
PROOFS FOR THE MAIN PAPER

Proof of Lemma 3

Proof. Let $\mathcal{V}_{\mathrm{l}}$ be all elements in $\mathcal{V}$ that do not exceed ξ.

Under the swap paradigm, the optimal strategy to maximally increase a set's median is to iteratively replace the set's smallest value with ∞. Apply this optimal strategy to $\mathcal{V}$. After one swap, the resulting set contains $|\mathcal{V}_{\mathrm{l}}| - 1$ elements that are less than or equal to ξ. After two swaps, there are $|\mathcal{V}_{\mathrm{l}}| - 2$ such elements with each subsequent swap's effects proceeding inductively. Once the modified set contains exactly $\left\lceil \frac{T}{2} \right\rceil$ elements less than or equal to ξ, no additional swaps are possible without causing the resulting set's median to exceed ξ.

Therefore, by induction, the maximum number of swaps that can be performed on $\mathcal{V}$ and it remains guaranteed that the resulting set's median does not exceed ξ is

$$R = |\mathcal{V}_{\mathrm{l}}| - \left\lceil \frac{T}{2} \right\rceil. \tag{20}$$

□

Proof of Lemma 4

Proof. For simplicity and w.l.o.g., assume $\mathcal{V}$ is ordered where $\nu_1 \leq \cdots \leq \nu_T$. An attacker's optimal insertion strategy is to insert arbitrarily-large values into $\mathcal{V}$ while the optimal deletion strategy is to always delete $\mathcal{V}$'s smallest value. Lemma 14 proves that these worst-case operations perturb the median identically so we only consider insertions below.

Let α denote the median's index. If T is odd, $\alpha = \left\lceil \frac{T}{2} \right\rceil$; otherwise, when T is even, $\alpha = \frac{T}{2} + \frac{1}{2}$, i.e., the midpoint between the $\frac{T}{2}$-th and $(\frac{T}{2} + 1)$-th largest values in $\mathcal{V}$.

Each insertion increases the set's size by 1. When $\mathcal{V}$'s size is odd, then the size of this new set after the first insertion is even. Therefore, this new set's median has index

$$\alpha' \coloneqq \frac{T+1}{2} + \frac{1}{2} \qquad \text{Median's index for new even size } T+1 \tag{21}$$

$$= \left\lceil \frac{T}{2} \right\rceil + \frac{1}{2} \tag{22}$$

$$= \alpha + \frac{1}{2}. \tag{23}$$

The analysis is essentially identical when T is even and is excluded for brevity. Note that each insertion always increases the median's index α by $\frac{1}{2}$.

As long as $\alpha \leq |\mathcal{V}_{\mathrm{l}}|$, it is guaranteed that $\operatorname{med} \mathcal{V} \leq \xi$. Since each insertion changes α by $\frac{1}{2}$, then $2(|\mathcal{V}_{\mathrm{l}}| - \alpha)$ arbitrary insertions can be made in $\mathcal{V}$ with it remaining guaranteed that the modified set's median does not exceed ξ. Regardless of whether T is odd or even,[11] it holds that

$$R \geq 2(|\mathcal{V}_{\mathrm{l}}| - \alpha) = 2|\mathcal{V}_{\mathrm{l}}| - 2\alpha = 2|\mathcal{V}_{\mathrm{l}}| - T - 1. \tag{24}$$

□

[11] When T is odd, $2\alpha = 2\left\lceil \frac{T}{2} \right\rceil = T + 1$ while in the even case $2\alpha = 2\left(\frac{T}{2} + \frac{1}{2}\right) = T + 1$.

Proof of Lemma 5

Proof. The first portion of this proof follows the same argument as the proof of Lemma 3, with one primary difference. There, the optimal strategy to perturb $\mathcal{V}$'s median swapped out the smallest values in $\mathcal{V}$ first. For the weighted version, the optimal (greedy) strategy swaps out whichever value in $\mathcal{V}_\text{l} \subseteq \mathcal{V}$ has the smallest weight.

To perturb $\mathcal{V}$'s median above ξ, it is sufficient to swap any $|\mathcal{V}_\text{l}| - \lceil \frac{T}{2} \rceil$ values in $\mathcal{V}_\text{l}$ with an arbitrarily large replacement. For simplicity and without loss of generality, let $\tilde{r}_1, \ldots, \tilde{r}_{|\mathcal{V}_\text{l}|}$ be the weights of the elements in $\mathcal{V}_\text{l}$ arranged in ascending order. Define $\Delta \coloneqq |\mathcal{V}_\text{l}| - \lceil \frac{T}{2} \rceil$. Applying Lemma 3, up to Δ values in $\mathcal{V}$ can be replaced without perturbing the median. This entails a minimum cost of

$$R \geq \sum_{t=1}^{\Delta} \tilde{r}_t. \tag{25}$$

Denote the $(\Delta+1)$-th largest weight in $\mathcal{R}_\text{l}$ as $\tilde{r}_{\Delta+1}$. Observe that adding $\tilde{r}_{\Delta+1} - 1$ to Eq. (25) is insufficient to swap out any remaining elements in $\mathcal{V}_\text{l}$ since all elements with weight less than $\tilde{r}_{\Delta+1}$ are already replaced and all remaining elements have weight at least $\tilde{r}_{\Delta+1}$. Therefore, the certified robustness is

$$\begin{aligned} R &= (\tilde{r}_{\Delta+1} - 1) + \sum_{t=1}^{\Delta} \tilde{r}_t && (26)\\ &= \sum_{t=1}^{\Delta+1} \tilde{r}_t - 1 && (27)\\ &= \sum_{t=1}^{|\mathcal{V}_\text{l}| - \lceil \frac{T}{2} \rceil + 1} \tilde{r}_t - 1 && (28)\\ &= \sum_{r \in \widetilde{\mathcal{R}}_\text{l}} r - 1. && (29) \end{aligned}$$

Moreover, increasing Eq. (29) by one would allow for the $(\Delta+1)$-th largest value in $\mathcal{V}$ to be swapped, which would in turn perturb the set's median above ξ. Therefore, Eq. (29)'s bound is tight. □

Proof of Lemma 6

Proof. The median perturbation paradigms formalized in Lemmas 3, 4, and 5 calculate their certified robustness using three values, namely: T, $|\mathcal{V}_\text{l}|$, and $\lceil \frac{T}{2} \rceil$. If these three values are equivalent for $\mathcal{V}$ and $\mathcal{V}_{\pm 1}$, then their associated certified robustness (R) must also be equal.

Since $|\mathcal{V}| = |\mathcal{V}_{\pm 1}|$, they have equivalent T and $\lceil \frac{T}{2} \rceil$. By definition, the binarization of $\mathcal{V}$ to $\mathcal{V}_{\pm 1}$ does not change the value of $|\mathcal{V}_\text{l}|$ either. Therefore, for all three median perturbation paradigms, binary multiset $\mathcal{V}_{\pm 1}$ and real multiset $\mathcal{V}$ have equivalent certified robustness R. □

Proof of Theorem 7

Proof. For fixed-population IBLs, certifying that $f(x_\text{te}) \leq \xi$ simplifies to median perturbation under Sec. IV-A's unweighted swap paradigm since all necessary criteria are met, namely that

1) f's decision function is a median operation over a set of values, i.e., $f(x_\text{te}) \coloneqq \text{med}\,\mathcal{N}(x_\text{te})$.
2) Neighborhood $\mathcal{N}(x_\text{te})$ has fixed cardinality T, and T is odd.
3) A worst-case modification to training set $\mathcal{S}$ causes an element in $\mathcal{N}(x_\text{te})$ to be replaced with a different one.

Lemma 3, therefore, provides a (lower) bound on the number of training set modifications that can be made without the resulting model violating the requirement that $f(x_\text{te}) \leq \xi$. That is why certified robustness R in Eqs. (6) and (2) (Thm. 7 & Lem. 3, resp.) are equivalent. □

Proof of Theorem 8

Proof. This proof follows a very similar structure as Theorem 7's proof above. The primary distinction is that a different median perturbation paradigm from Sec. IV is needed here.

For region-based IBLs, certifying that $f(x_{\text{te}}) \leq \xi$ simplifies to median perturbation under Sec. IV-B's insertion/deletion paradigm since the three necessary criteria are met:

1) f's decision function is a median operation over a set of values, i.e., $f(x_{\text{te}}) \coloneqq \text{med}\,\mathcal{N}(x_{\text{te}})$.
2) Neighborhood cardinality T is not fixed but can increase and/or decrease.
3) Each modification of $\mathcal{N}(x_{\text{te}})$ takes the form of either an insertion or deletion, i.e., not swaps.

Therefore, Lemma 4 bounds the total number of training set insertions/deletions that can be performed without violating the requirement that $f(x_{\text{te}}) \leq \xi$. That is why R's definition in Eq. (7) is identical to Eq. (3). □

Proof of Theorem 9

Proof. Certifying here that $f(x_{\text{te}}) \leq \xi$ simplifies to median perturbation under the unweighted swap paradigm since all necessary criteria are satisfied, specifically that

1) f's decision function is a median operation over a set of fixed, deterministic values, i.e., $f(x_{\text{te}}) \coloneqq \text{med}\,\{f_1(x_{\text{te}}), \ldots, f_T(x_{\text{te}})\}$.
2) Since the submodels are trained on disjoint data/feature regions, a change to one submodel (i.e., value $f_t(x_{\text{te}})$) has no effect on any other submodel (value).
3) Each submodel perturbation causes an existing value in the set to be replaced by a new value.
4) T is fixed and odd-valued.
5) The cost to change any submodel (i.e., value) is one, i.e., $\forall_t\, r_t = 1$.

Lemma 3 provides a (lower) bound on the number of training set modifications that can be performed without violating the requirement that $f(x_{\text{te}}) \leq \xi$. Certified robustness R in Eq. (8) is then identical to Lemma 3's Eq. (2). □

Proof of Theorem 10

Proof. Here, we extend the argument in Theorem 9's proof to the weighted case. Four of the five criteria in Thm. 9's proof still hold, specifically that

1) f's decision function is a median over a set of values.
2) Each submodel is independent and deterministic.
3) Modifications to the set of values take the form of swaps.
4) T is fixed and odd-valued.

The only difference is that the perturbations are weighted where each value $f_t(x_{\text{te}})$ now has an associated cost $r_t \geq 0$. Therefore, Sec. IV-C's weighted swap paradigm applies. Certified robustness R in Eq. (9) follows directly from and is identical to R in Lemma 5's Eq. (5). □

Proof of Lemma 11

Proof. To prove a problem is NP-hard, it suffices to show that there exists a polynomial time reduction from a known NP-hard problem to it. As explained in Sec. VIII-A, partial set cover is NP-hard [48, 49]. Below we map partial set cover to overlapping certified regression.

Let $U \coloneqq [T']$ be a ground set of T' elements, and let $\mathbf{Q} \coloneqq \{\mathcal{Q}_1, \ldots, \mathcal{Q}_m\}$ be a collection of sets where each $\mathcal{Q}_j \subseteq U$ and

$$\bigcup_{\mathcal{Q} \in \mathbf{Q}} \mathcal{Q} = U.$$

The goal is to find the subcover $\mathcal{F} \subseteq \mathbf{Q}$ of minimum cardinality s.t.

$$\Delta \leq \left| \bigcup_{\mathcal{Q} \in \mathcal{F}} \mathcal{Q} \right|,$$

where $\Delta \in [T']$.

It is straightforward to map the above to overlapping certified regression. Let the ensemble have $(2T'+1)$ submodels. Function h_{tr} partitions the training set into m blocks with the blocks denoted $S^{(1)}, \ldots, S^{(m)}$. Define the block mapping function as:

$$h_f(j) \coloneqq \begin{cases} \mathcal{Q}_j, & j \leq T' \\ \emptyset, & \text{Otherwise} \end{cases}. \tag{30}$$

Intuitively, each of the first T' submodels is trained on one of the subsets in $\mathbf{Q}$, while the remaining models are not trained on any data.

Let all submodels be constant a function. Define the submodel function as

$$f_t(x_{\text{te}}) \coloneqq \begin{cases} -\infty, & t \leq \Delta + \left\lceil \frac{T}{2} \right\rceil \\ \infty, & \text{Otherwise} \end{cases}. \tag{31}$$

For any finite ξ, $|\mathcal{V}_{\text{l}}| = \Delta + \left\lceil \frac{T}{2} \right\rceil$. Applying Theorem 9, the number of submodels overlapping certified regression perturbs is $|\mathcal{V}_{\text{l}}| - \left\lceil \frac{T}{2} \right\rceil = \Delta$.

Overlapping certified regression's robustness R is the solution to the original partial set cover problem because

1) Only models with index $t \leq T'$ will be perturbed since all other submodels have no training data.
2) The training set of each of these T' submodels maps directly to a subset in $\mathbf{Q}$.
3) Overlapping certified regression seeks to find the minimum number of dataset blocks that must be modified to perturb the median prediction. In this formulation, the number of blocks to be modified is Δ – same as in the original partial-set cover problem.

If overlapping certified regression were solvable in polynomial-time, then partial set cover would also be solvable in polynomial time. However, partial set cover is NP-hard, meaning overlapping certified regression must also be NP-hard. □

Proof of Corollary 12

Proof. From Lemma 11 above, (unweighted) overlapping certified regression is NP-hard. The unweighted case trivially maps to the weighted one where $\forall_t \, r_t = 1$. Therefore, weighted OCR must be at least as hard as the unweighted case meaning W-OCR is also NP-hard. □

Proof of Lemma 13

Proof. By construction

Given a deterministic training algorithm, a model's prediction can be certified against the deletion of any subset $S \subset \mathcal{S}$ by training a model on just dataset $S \setminus S$ and verifying the prediction does not violate the associated *invariant*, i.e., $f_{\mathcal{S} \setminus S}(x_{\text{te}}) \leq \xi$.

Consider training a separate model on each subset of $\mathcal{S}$ of size at least $n - r + 1$. If *all* of those models also satisfy the invariant, then by construction, $r - 1$ deletions or fewer are insufficient to violate the invariant. If $r - 1$ deletions are not enough, then at least r deletions are required. □

Lemma 13's proof above only applies if model prediction and training is deterministic, i.e., repeating training and then the prediction always yields the same predicted value. Otherwise, proof by construction would require verifying all random seeds for each subset of $\mathcal{S}$.

APPENDIX D
REDUCING kNN-CR TO JIA ET AL.'S CERTIFIED kNN CLASSIFIER

Sec. III mentions that Jia et al. [24] propose a certified classifier by leveraging the implicit robustness of nearest neighbor methods. In their paper, Jia et al. examine both *pointwise certification*[12] of test instances individually as well as *joint certification* of multiple test instances collectively. Here, we exclusively consider Jia et al.'s pointwise contributions.

This section explores in detail how our certified robust regressor, kNN-CR, reduces to Jia et al.'s certified kNN classifier. For classification, label space $\mathcal{Y}$ is nominal and consists of $|\mathcal{Y}|$ classes. Given test instance $x_{\text{te}} \in \mathcal{X}$, a kNN classifier returns neighborhood $\mathcal{N}(x_{\text{te}})$, which is a multiset whose *underlying set* is $\mathcal{Y}$. Let c and c' be the labels with the largest and second-largest *multiplicity* in $\mathcal{N}(x_{\text{te}})$. In other words, c and c' are the first and second most popular labels in x_{te}'s neighborhood. Denote the multiplicity of labels c and c' as T_c and $T_{c'}$, respectively.

Jia et al. [24, Thm. 1] specify their certified kNN classifier's robustness bound as

$$R = \left\lceil \frac{T_c - T_{c'} + \mathbb{1}[c > c']}{2} \right\rceil - 1, \tag{32}$$

where indicator function $\mathbb{1}[c > c']$ breaks ties by choosing whichever label is assigned the larger number.

Lemma 15 below establishes that for binary classification, Jia et al.'s method and kNN-CR certify equivalent robustness guarantees (R). This symmetry between Jia et al.'s method and kNN-CR extends beyond classification to regression as well.

Recall that Lemma 6 establishes symmetry between certified robustness in a real-valued domain and robustness in a binarized domain. Applying that insight here, Jia et al.'s certified kNN (binary) classifier is extended from the binary domain to certify the robustness of kNN-CR's real-valued regression predictions. Observe that this interplay is the fundamental concept underpinning any reduction. Put simply, Jia et al.'s certification algorithm serves as, in essence, a "subroutine" within Fig. 3's certified regressor framework.

Lemma 15. *For $x_{te} \in \mathcal{X}$ and f a k-nearest neighbor classifier, let $\mathcal{N}(x_{te})$ be x_{te}'s neighborhood under f with $T \coloneqq |\mathcal{N}(x_{te})|$ odd. For binary classification, Jia et al. [24] and kNN-CR certify equivalent robustness.*

Proof. For binary classification with odd neighborhood cardinality T, there cannot be ties between labels. Moreover, $T_c - T_{c'}$ is always odd by the pigeonhole principle. Applying these two observations, Eq. (32) simplifies to

$$R = \left\lceil \frac{T_c - T_{c'}}{2} \right\rceil - 1 \qquad \triangleright \text{ No ties} \tag{33}$$

$$= \frac{T_c - T_{c'} + 1}{2} - 1. \qquad \triangleright\ T_c - T_{c'} \text{ is odd} \tag{34}$$

Under binary classification, $T_c + T_{c'} = T$ meaning

$$R = \frac{T_c - (T - T_c) + 1}{2} - 1 \tag{35}$$

$$= \frac{2T_c - T - 1}{2} \tag{36}$$

$$= T_c - \frac{T+1}{2} \tag{37}$$

$$= T_c + \left\lceil \frac{T}{2} \right\rceil. \tag{38}$$

In the case of binary classification, $\mathcal{V}_1$ represents all models that predict majority label c. Therefore, $|\mathcal{V}_1| = T_c$ meaning Eq. (38) is equivalent to Eq. (6). □

[12] Jia et al. [24] term "pointwise certification" as "*individual certification*."

APPENDIX E
REDUCING PCR TO DPA

As detailed in Sec. III, Levine and Feizi [25]'s deep partition aggregation (DPA) defense relies on an ensemble of T fully-independent, deterministic submodels,[13] each of which is trained on disjoint data. Like our certified regressors, DPA certifies the *pointwise* robustness of an individual model prediction, $f(x_{\text{te}})$.

In the classification case, label space $\mathcal{Y}$ is nominal and consists of $|\mathcal{Y}|$ classes. Given $x_{\text{te}} \in \mathcal{X}$, each submodel assigns x_{te} some label, i.e., $f_t(x_{\text{te}}) \in \mathcal{Y}$. Denote the ensemble's plurality label for x_{te} as $c \in \mathcal{Y}$, and denote the number of submodels that predict c for x_{te} as $T_c \in [T]$. Let $c' \in \mathcal{Y} \setminus \{c\}$ denote any label other than the ensemble's plurality prediction, and let $T_{c'} \in \{0, \dots, \lceil \frac{T}{2} \rceil\}$ denote the number of submodels that predict c' for x_{te}.

Levine and Feizi [25, Thm. 1] specify DPA's certified robustness bound as

$$R = \left\lfloor \frac{T_c - \max_{c' \neq c} \{T_{c'} + \mathbb{1}[c' < c]\}}{2} \right\rfloor, \tag{39}$$

where $\mathbb{1}[c' < c]$ breaks ties by predicting whichever label has the lower assigned number.

Lemma 16 below establishes that DPA and PCR certify equivalent robustness guarantees for binary classification with an odd number of submodels. This symmetry between DPA and PCR extends beyond classification to regression as well.

Recall that Lemma 6 establishes symmetry between certified robustness in a real-valued domain and robustness in a binarized domain. Applying that insight, it is clear then that DPA is certifying the robustness of our certified regressor PCR, albeit using $\mathcal{V}$'s surrogate $\mathcal{V}_{\pm 1} \coloneqq \{\operatorname{sgn}(\nu_t - \xi) : \nu_t \in \mathcal{V}\}$. This is the fundamental idea behind reductions. Here, DPA's certification algorithm serves as a type of "subroutine" within the certified regressor framework (Fig. 3).

Lemma 16. *Let f be an ensemble of T fully-independent, deterministic submodels where each submodel is trained on disjoint data with T odd. For binary classification, DPA and* PCR *certify equivalent robustness.*

Proof. This proof is very similar to that of Lem. 15. Nonetheless, we repeat the full details to make each proof standalone.

Let $x_{\text{te}} \in \mathcal{X}$ be any feature vector. Given ensemble f, x_{te}'s non-majority label c' is unique so Eq. (39) simplifies to

$$R = \left\lfloor \frac{T_c - (T_{c'} + \mathbb{1}[c' < c])}{2} \right\rfloor. \tag{40}$$

Similarly for odd T, ties are not possible making the indicator function unnecessary and able to be ignored depending how c and c' are chosen to be defined. This simplifies Eq. (40) to

$$R = \left\lfloor \frac{T_c - T_{c'}}{2} \right\rfloor. \tag{41}$$

Eq. (41) can be rewritten as

$$\begin{aligned}
R &= \frac{T_c - T_{c'} - 1}{2} && \triangleright\ T_c - T_{c'} \text{ always odd when } T \text{ is odd} && (42)\\
&= \frac{T_c - (T - T_c) - 1}{2} && \triangleright\ T_c + T_{c'} = T \text{ for binary classification} && (43)\\
&= T_c - \frac{T+1}{2} && && (44)\\
&= T_c - \left\lceil \frac{T}{2} \right\rceil. && && (45)
\end{aligned}$$

In the case of binary classification, $\mathcal{V}_{\text{l}}$ represents all models that predict majority label c. Therefore, $|\mathcal{V}_{\text{l}}| = T_c$ meaning Eq. (45) is equivalent to Eq. (8). □

[13] Levine and Feizi do not specify that T is odd, but such a choice is standard in binary classification settings.

APPENDIX F
RELATING OCR TO WANG ET AL.'S OVERLAPPING CERTIFIED CLASSIFIER

As discussed in Sec. III, Wang et al. [26] propose an overlapping certified classifier they name *deterministic finite aggregation* (DFA). Note the acronym similarity between DFA and Levine and Feizi's [25] deep partition aggregation (DPA).

As a basic intuition on DFA, recall that $d_{\max}$ is the maximum spread degree of any dataset block over the T overlapping submodels. As detailed in Sec. VIII, when certifying robustness for both classification and regression, (partially) covering set $[T]$ is *not* the goal. Rather, the goal is to partially cover set

$$\mathcal{T}_{\mathrm{l}} \coloneqq \{t : f_t(x_{\mathrm{te}}) \leq \xi\} \subseteq [T], \tag{46}$$

i.e., those subset of models that predict at most ξ. Over this restricted subset, the maximum block spread degree may be less than $d_{\max}$. As a very simplified explanation, Wang et al. use this insight to provide worst-case deterministic bounds on the robustness of an overlapping classification prediction.

Our discussion of overlapping regression in Sec. VIII does not directly apply Wang et al.'s robustness certifier for multiple reasons, including

1) Wang et al. do not explain that determining optimal R under their formulation is NP-hard. We believe this is an important insight.
2) Wang et al. do not analyze the optimality gap of their approach. Both our ILP approach and partial set cover approximation give optimality bounds. What is more, our ILP-based approach can actually prove optimality gives $r_1, \ldots, r_T$. In contrast, when Wang et al.'s solution is optimal, no indication of that fact is readily given.
3) Wang et al. only consider the unit-cost assumption, meaning their formulation does not natively support weighted robustness analysis. To ensure a fair and direct comparison between OCR and W-OCR, we use an ILP for both methods.

One clear limitation of our choice to use an ILP for overlapping robustness certification is the added computational cost. However, in our implementation, solving a single ILP uses very few resources, e.g., just a single core in a multicore CPU. Moreover, we enforce tractable computation times by specifying an ILP time limit. While these two factors combined are not a panacea, they do assuage (some) computational concerns.

APPENDIX G
ON THE TIGHTNESS OF CERTIFIED REGRESSION

This section explores two cases where our certified robustness bounds may not be tight.

A. Region-Based Neighborhood IBL

Recall that our insertion/deletion paradigm specifies the certified robustness as

$$R = 2|\mathcal{V}_\text{l}| - T - 1.$$

Consider the case where $|\mathcal{V}_\text{l}| < T$. After R worst-case insertions into set $\mathcal{V}$, the median is $\frac{\max \mathcal{V}_\text{l} + \min \mathcal{V}_\text{u}}{2}$. It is possible that ξ can still exceed this value, meaning one more worst-case insertion/deletion is possible. In other words, Eq. (3)'s bound would be non-tight by at most one.

To make the insertion/deletion paradigm's certified robustness bound tight, redefine Lemma 4's certified robustness as,

$$R = 2|\mathcal{V}_\text{l}| - T - 1 + \mathbb{1}\left[\frac{\max \mathcal{V}_\text{l} + \min \widetilde{\mathcal{V}}_\text{u}}{2} \leq \xi\right], \tag{47}$$

where $\widetilde{\mathcal{V}}_\text{u} \coloneqq \mathcal{V}_\text{u} \cup \{\infty\}$.[14]

Since Theorem 8 derives its bounds directly from Lemma 4, our certified robustness bound for region-based neighborhood IBLs can also be non-tight by one. We deliberately simplify our formulation to exclude this case to ensure consistency with the other presented ideas.

B. Weighted Overlapping Certified Regression

A weighted ensemble never has a worse *true* certified robustness than its unit-cost equivalent. However, there are cases where W-OCR's bound is one lower than OCR's bound, i.e., Fig. 6's ILP bound is not tight. This occurs, when for optimal R, it is possible to perturb $|\mathcal{V}_\text{l}| - \lceil \frac{T}{2} \rceil + 1$ submodels (i.e., one more than the required $|\mathcal{V}_\text{l}| - \lceil \frac{T}{2} \rceil$ submodels). In such cases, Eq. (5)'s decrementing by one causes the bound to be one less than the ideal value. It is possible to formulate a more complicated ILP to account for this corner case. However, we deliberately keep the formulation simple, knowing R may be marginally non-tight. Sec. X's evaluation shows this niche case occurs only rarely in practice – almost exclusively when T is small.

[14] Including ∞ addresses an edge case in Eq. (47) when $\mathcal{V} = \mathcal{V}_\text{l}$, and $\min \emptyset$ is undefined. When $\mathcal{V}_\text{u} \neq \emptyset$, the ∞ has no effect.

APPENDIX H
EVALUATION SETUP

This section details the evaluation setup used in Section X's experiments, including implementation details, dataset configuration, and hyperparameter settings.

Our source code can be downloaded from https://github.com/ZaydH/certified-regression. All experiments were implemented and tested in Python 3.7.1. Experiments were performed using one core of a fourteen-core Intel E5-2690v4 CPU and 12GB of RAM. Ridge regression models were trained using Scikit-Learn [65], while the decision forests used the XGBoost library [21].

The overlapping regressor ILPs (Fig. 6) were optimized using Gurobi [54] with a time limit of 1200s. Our implementation loads the `gurobipy` Gurobi python package by default. Gurobi is a commercial product and requires a license to solve most non-trivial linear programs. Gurobi offers free, unlimited licenses for academic use, including both for an individual system and for a cloud/HPC environment.

A. Dataset Configuration

Our source code automatically downloads all necessary datasets.

Regarding dataset preprocessing, categorical features were transformed into one-hot-encoded features in line with previous work [39]. Standardizing features by dataset mean/variance breaks submodel independence and so was not performed. Minimal manual feature engineering was performed to improve the housing datasets' results, e.g., adding a home's age, total square feet, total number of bathrooms, etc.; this feature engineering was done based on existing features in the dataset (e.g., total square feet equals the sum of the first and second-floor square footage). None of the engineered features affect submodel independence.

Most of the six datasets in Sec. X-A do not have a dedicated test set. In such cases, the data was split 90%/10% at random between training and test.

When training kNN-CR models, each feature dimension was normalized to the range $[0, 1]$. Without feature normalization, kNN-CR generally prioritizes whichever feature has the largest magnitude. This transformation implicitly restricts arbitrary insertions to the feature range in the original dataset. Such normalization is implicitly done in certified classifier evaluation on image datasets where each pixel has a consistent, fixed range.

B. Dataset Target Value Statistics

Table III summarizes the test set's target (y) value distribution statistics for Sec. X's five regression datasets.

Recall from Table I that the Ames, Austin, and Diamonds datasets set error threshold ξ as a fixed percentage of y_{te}. This choice was made because these three datasets exhibit significant y variance. For example, for Diamonds, the largest y value ($18.8k) is about two orders of magnitude larger than the smallest y value ($339). Using a fixed ξ value on these three datasets would have made certifying instances with small y unrealistically easy while making certification of instances with large y unreasonably difficult. Making the error threshold a fraction of y_{te} allows the certification difficulty to be more consistent across the range of y values.

Datasets Weather and Life used fixed ξ values of 3 degrees (Celsius) and 3 years respectively. Both of these threshold values are less than one-third of each dataset's y standard deviation.

Supplemental Section I-C evaluates the performance of our certified regressors on additional ξ values – both larger and smaller than the ξ values used in Sec. X.

C. Hyperparameters

Following Jia et al.'s [24] certified kNN classifier evaluation, kNN-CR's neighborhood size, k, was set to the (larger) odd integer nearest to $\frac{n}{2}$. We use the Minkowski distance as the neighborhood's distance metric.

For our ensemble regressors, hyperparameters were tuned using Bayesian optimization as implemented in the `scikit-optimize` library [66]. The partitioned and overlapping certified regressors (unweighted and weighted) used the same hyperparameter settings.

TABLE III: **Target Value Test Distribution Statistics**: Mean ($\bar{y}$), standard deviation (σ_y), minimum value ($y_{\min}$) and maximum value ($y_{\max}$) for the test instances' target y value for Sec. X's five regression datasets.

Dataset	$\bar{y}$	σ_y	$y_{\min}$	$y_{\max}$
Ames	\$184k	\$83.4k	\$12.8k	\$585k
Austin	\$466k	\$266k	\$81.0k	\$2.6M
Diamonds	\$3.8k	\$3.9k	\$0.3k	\$18.8k
Weather	14.9°C	10.3°C	−44.0°C	54.0°C
Life	69.3 years	9.6 years	36.3 years	89.0 years

Ridge Regression Hyperparameters For three datasets – Diamonds [60], Weather [61], and Spambase [63] – our four ensemble regressors used ridge regression as the submodel architecture. For each dataset and q value, we tuned three ridge regression hyperparameters. Below, we list those hyperparameters along with the set of values considered.

- *Weight Decay* (λ): L_2 regularization strength. We considered values between 10^{-8} and 10^{4}.
- *Error Tolerance* (ε): Minimum validation error that defines when a model is considered converged. The tested values were $\{10^{-8}, 10^{-7}, \ldots, 10^{-3}\}$.
- *Maximum Number of Iterations* (# Itr.): Defines the maximum number of optimizer iterations. If the error tolerance is achieved before the iteration count is met, the model is treated as converged, and optimization stops. The tested values were $\{10^{2}, 10^{3}, \ldots, 10^{8}\}$.

Table IV lists the final hyperparameters for each experimental setup that used ridge regression as the submodel architecture.

XGBoost Hyperparameters For three datasets – Ames Housing [58], Austin Housing [59], and Life [62] – our four ensemble regressors used XGBoost [21] as the submodel architecture. For each dataset and q value, we tuned seven XGBoost hyperparameters. Below, we list those hyperparameters along with the set of values considered.

- *Number of Trees* (τ): Number of trees in the ensemble. The tested values were $\{50, 100, 250, 500, 1000\}$.
- *Maximum Tree Depth* (h): Maximum depth of each tree in the ensemble. The tested values were $\{1, \ldots, 4\}$.
- *Evaluation Metric* ($\mathcal{L}$): Applied to the validation set and is the metric being minimized. The tested values were root mean squared error (RMSE) and mean absolute error (MAE).
- *Weight Decay* (λ): L_2 regularization strength. We considered values between 10^{-3} and 10^{5}.
- *Minimum Split Loss* (γ): Minimum reduction in loss required to split a node instead of making it a leaf. The values considered were $\{0.005, 0.01, 0.05, 0.1, 0.3, 0.5, 1\}$.
- *Learning Rate* (η): Larger value makes the boosting more conservative. The tested values were $\{0.01, 0.1, 0.3, 1\}$.

Table V lists the final hyperparameters for each experimental setup that used XGBoost as the submodel architecture. Mixup [64] data augmentation was used to improve XGBoost's performance.[15]

[15]Mixup does not apply to convex models like ridge regression.

TABLE IV: **Ridge Regression Hyperparameters**: Hyperparameter settings for the three datasets that used ridge regression as the ensemble submodel architecture. Hyperparameters are reported for the three q values used in Fig. 7 and Sec. I-A. We also report the hyperparameters for uncertified accuracy when $q = 1$.

Dataset	q	λ	ε	# Itr.
Diamonds	1	3.16E−3	1E−6	1E6
	151	6.01E−2	1E−7	1E8
	501	1.00E−8	1E−6	1E8
	1001	1.38E−8	1E−6	1E2
Weather	1	3.16E−3	1E−8	1E7
	51	1.00E+3	1E−5	1E6
	1501	3.16E+2	1E−6	1E2
	3001	3.16E+2	1E−6	1E3
Spambase	1	3.16E+2	1E−6	1E5
	25	3.16E−3	1E−6	1E6
	151	3.16E−6	1E−7	1E6
	301	3.16E−3	1E−6	1E6

TABLE V: **XGBoost Hyperparameters**: Hyperparameter settings for the three datasets that used XGBoost as the ensemble submodel architecture. Hyperparameters are reported for the three q values used in Fig. 7 and Sec. I-A. We also report the hyperparameters for uncertified accuracy when $q = 1$.

Dataset	q	τ	h	$\mathcal{L}$	λ	γ	η
Ames Housing	1	250	2	RMSE	1E−1	5E−3	0.3
	25	500	2	MAE	1E−3	5E−3	0.3
	125	500	3	RMSE	1E−2	5E−3	1.0
	251	250	1	RMSE	1E−1	5E−3	1.0
Austin Housing	1	500	4	MAE	1E+2	1E−2	0.3
	151	1000	1	RMSE	1E−2	5E−3	1.0
	301	250	1	MAE	1E+0	1E−2	1.0
	701	250	1	MAE	1E−2	1E−2	1.0
Life	1	500	5	RMSE	1E+1	1E−2	0.1
	25	250	4	RMSE	0E+0	5E−2	0.3
	101	250	3	MAE	1E+0	1E−2	1.0
	201	250	4	RMSE	0E+0	5E−3	0.3

APPENDIX I
ADDITIONAL EXPERIMENTS

Limited space prevents us from including all experimental results in the main paper. We provide additional results below.

A. Detailed Experimental Results

1) Baseline Accuracy: Table VI shows the baseline accuracy when a model is trained on all of training set $\mathcal{S}$ (i.e., $q = 1$). For each dataset, the model architecture (either ridge regression or XGBoost) aligns with those used for Sec. X's ensembles. See Table I.

TABLE VI: **Baseline Accuracy**: Summary of the baseline (i.e., uncertified) accuracy mean and standard deviation for Sec. X's six datasets. Submodels were trained on all of training set $\mathcal{S}$ (i.e., $q = 1$). Beside each dataset's name is the submodel architecture used by the ensemble. Threshold ξ matches values in Table I.

Dataset	Submodel	Base Acc. (%)
Ames	XGBoost	90.4 ± 2.4
Austin	XGBoost	71.3 ± 4.1
Diamonds	Ridge	73.6 ± 4.0
Weather	Ridge	85.9 ± 3.4
Life	XGBoost	92.7 ± 3.1
Spambase	Ridge	87.5 ± 2.9

2) Numerical Results: Fig. 7 visualizes our certified regressors' certified robustness on six datasets – five regression and one binary classification. This section provides the certified accuracy in numerical form, including the associated variance.

TABLE VII: **Ames Housing Full Results**: Certified accuracy mean and standard deviation for the Ames Housing [58] dataset. Each ensemble submodel was trained on $\frac{1}{q}$-th of the training set with three q values tested per dataset, while kNN-CR was always trained on the whole training set (i.e., $q = 1$). The certified accuracy results of five robustness values (R) are reported per q value. Also reported as a baseline is the uncertified accuracy ($R = 0$) when training a single model on all of training set $\mathcal{S}$ ($q = 1$). Results are averaged across 10 trials per method, with each R's best mean certified accuracy in bold.

q	R	PCR	OCR	W-PCR	W-OCR	kNN-CR
1	0	90.4 ± 2.4	90.4 ± 2.4	90.4 ± 2.4	90.4 ± 2.4	54.3 ± 3.8
25	1	82.3 ± 3.1	**86.8 ± 2.8**	82.3 ± 3.1	85.2 ± 2.9	54.3 ± 3.8
	4	76.3 ± 3.7	**81.2 ± 2.9**	76.3 ± 3.7	80.0 ± 2.7	54.0 ± 3.6
	8	57.4 ± 3.8	69.6 ± 2.7	57.5 ± 3.7	**70.1 ± 3.2**	53.1 ± 3.5
	12	11.0 ± 3.7	28.2 ± 3.8	23.7 ± 5.1	48.8 ± 4.4	**52.2 ± 3.9**
	16	0.0 ± 0.0	0.0 ± 0.0	11.3 ± 3.8	15.3 ± 3.2	**51.5 ± 3.8**
125	1	70.4 ± 3.7	**73.9 ± 1.7**	70.4 ± 3.7	73.1 ± 1.6	54.3 ± 3.8
	10	62.8 ± 3.4	**66.5 ± 2.0**	62.8 ± 3.4	65.9 ± 1.9	53.1 ± 3.5
	20	44.9 ± 4.2	**52.2 ± 2.9**	44.9 ± 4.2	**52.2 ± 3.0**	51.2 ± 3.6
	30	21.8 ± 4.2	28.7 ± 3.8	21.8 ± 4.2	31.1 ± 3.3	**49.1 ± 3.7**
	40	2.5 ± 1.3	3.9 ± 1.7	2.5 ± 1.3	5.9 ± 2.3	**48.5 ± 4.0**
251	1	63.1 ± 3.4	**66.3 ± 3.8**	63.1 ± 3.4	66.0 ± 3.7	54.3 ± 3.8
	20	51.8 ± 3.2	56.4 ± 2.9	51.8 ± 3.2	**57.9 ± 2.9**	51.2 ± 3.6
	40	37.1 ± 3.2	42.5 ± 3.8	37.1 ± 3.2	45.3 ± 2.8	**48.5 ± 4.0**
	60	15.3 ± 3.8	22.5 ± 3.4	15.3 ± 3.8	32.8 ± 3.7	**44.1 ± 4.3**
	80	0.2 ± 0.4	0.6 ± 0.5	0.2 ± 0.4	10.9 ± 2.5	**40.1 ± 3.9**

TABLE VIII: **Austin Housing Full Results**: Certified accuracy mean and standard deviation for the Austin Housing [59] dataset. Each ensemble submodel was trained on $\frac{1}{q}$-th of the training set with three q values tested per dataset, while kNN-CR was always trained on the whole training set (i.e., $q = 1$). The certified accuracy results of five robustness values (R) are reported per q value. Also reported as a baseline is the uncertified accuracy ($R = 0$) when training a single model on all of training set $\mathcal{S}$ ($q = 1$). Results are averaged across 10 trials per method, with each R's best mean certified accuracy in bold.

q	R	PCR	OCR	W-PCR	W-OCR	kNN-CR
1	0	71.3 ± 4.1	71.3 ± 4.1	71.3 ± 4.1	71.3 ± 4.1	35.3 ± 4.8
51	1	59.9 ± 4.5	**63.7 ± 4.6**	59.9 ± 4.5	61.7 ± 4.6	35.3 ± 4.8
	5	49.6 ± 5.1	**52.9 ± 3.4**	49.6 ± 5.1	50.8 ± 4.2	35.2 ± 4.8
	10	29.9 ± 3.7	**35.3 ± 2.8**	29.9 ± 3.7	31.8 ± 3.2	34.9 ± 4.9
	15	9.2 ± 2.0	12.6 ± 2.8	9.2 ± 2.0	10.1 ± 2.7	**34.7 ± 4.9**
	20	0.5 ± 0.5	0.3 ± 0.7	0.5 ± 0.5	0.0 ± 0.0	**34.6 ± 4.6**
301	1	**51.0 ± 3.9**	52.0 ± 4.1	**51.0 ± 3.9**	51.1 ± 4.1	35.3 ± 4.8
	20	41.4 ± 3.6	**43.3 ± 5.9**	41.4 ± 3.6	**43.3 ± 5.9**	34.6 ± 4.6
	40	29.7 ± 4.1	32.2 ± 5.2	29.7 ± 4.1	33.7 ± 5.7	**34.3 ± 4.7**
	60	15.4 ± 3.0	19.1 ± 4.1	15.4 ± 3.0	22.7 ± 4.9	**34.0 ± 4.5**
	80	3.2 ± 1.8	3.1 ± 1.2	3.2 ± 1.8	7.7 ± 3.4	**32.9 ± 4.5**
701	1	**43.9 ± 5.0**	42.7 ± 5.5	**43.9 ± 5.0**	43.6 ± 5.7	35.3 ± 4.8
	40	34.5 ± 6.0	35.0 ± 6.2	34.5 ± 6.0	**36.9 ± 5.9**	34.3 ± 4.7
	80	25.3 ± 4.8	24.7 ± 6.1	25.3 ± 4.8	27.0 ± 6.1	**32.9 ± 4.5**
	120	13.1 ± 2.6	14.6 ± 5.0	13.1 ± 2.6	18.9 ± 4.4	**31.6 ± 4.9**
	160	2.7 ± 0.9	4.8 ± 3.3	2.7 ± 0.9	9.1 ± 2.9	**30.0 ± 4.7**

TABLE IX: **Diamonds Full Results**: Certified accuracy mean and standard deviation for the Diamonds [60] dataset. Each ensemble submodel was trained on $\frac{1}{q}$-th of the training set with three q values tested per dataset, while kNN-CR was always trained on the whole training set (i.e., $q = 1$). The certified accuracy results of five robustness values (R) are reported per q value. Also reported as a baseline is the uncertified accuracy ($R = 0$) when training a single model on all of training set $\mathcal{S}$ ($q = 1$). Results are averaged across 10 trials per method, with each R's best mean certified accuracy in bold.

q	R	PCR	OCR	W-PCR	W-OCR	kNN-CR
1	0	73.6 ± 4.0	73.6 ± 4.0	73.6 ± 4.0	73.6 ± 4.0	15.7 ± 3.5
151	1	74.6 ± 3.8	**74.8 ± 4.5**	74.6 ± 3.8	74.7 ± 4.4	15.7 ± 3.5
	35	64.4 ± 4.6	67.1 ± 4.8	67.2 ± 4.6	**69.7 ± 4.1**	15.6 ± 3.4
	70	38.6 ± 5.7	42.2 ± 4.0	62.4 ± 4.3	**64.7 ± 5.2**	15.5 ± 3.4
	105	0.0 ± 0.0	0.0 ± 0.0	54.5 ± 5.9	**57.4 ± 5.2**	15.2 ± 3.3
	140	0.0 ± 0.0	0.0 ± 0.0	**35.4 ± 5.8**	34.3 ± 7.1	14.8 ± 3.4
501	1	**77.3 ± 4.2**	75.8 ± 3.9	**77.3 ± 4.2**	75.7 ± 4.1	15.7 ± 3.5
	75	66.2 ± 4.0	65.0 ± 4.4	66.3 ± 4.0	**68.7 ± 4.4**	15.5 ± 3.4
	150	50.7 ± 4.9	48.2 ± 4.5	57.8 ± 4.7	**59.6 ± 4.9**	14.8 ± 3.4
	300	0.0 ± 0.0	0.0 ± 0.0	**38.0 ± 6.1**	36.2 ± 3.2	12.3 ± 3.4
	450	0.0 ± 0.0	0.0 ± 0.0	8.8 ± 3.3	9.0 ± 2.1	**10.7 ± 2.9**
1001	1	**75.2 ± 4.1**	74.9 ± 5.5	**75.2 ± 4.1**	74.9 ± 5.5	15.7 ± 3.5
	150	56.0 ± 4.9	56.3 ± 5.8	56.0 ± 4.9	**62.8 ± 6.1**	14.8 ± 3.4
	300	24.7 ± 4.4	25.3 ± 4.8	29.5 ± 4.1	**42.3 ± 6.4**	12.3 ± 3.4
	450	0.0 ± 0.0	0.0 ± 0.0	16.9 ± 4.0	**17.9 ± 5.1**	10.7 ± 2.9
	600	0.0 ± 0.0	0.0 ± 0.0	4.2 ± 1.5	3.3 ± 3.1	**9.6 ± 3.1**

TABLE X: **Weather Full Results**: Certified accuracy mean and standard deviation for the Weather [61] dataset. Each ensemble submodel was trained on $\frac{1}{q}$-th of the training set with three q values tested per dataset, while kNN-CR was always trained on the whole training set (i.e., $q = 1$). The certified accuracy results of five robustness values (R) are reported per q value. Also reported as a baseline is the uncertified accuracy ($R = 0$) when training a single model on all of training set $\mathcal{S}$ ($q = 1$). Results are averaged across 10 trials per method, with each R's best mean certified accuracy in bold.

q	R	PCR	OCR	W-PCR	W-OCR	kNN-CR
1	0	85.9 ± 3.4	85.9 ± 3.4	85.9 ± 3.4	85.9 ± 3.4	23.8 ± 4.5
51	1	86.0 ± 3.1	86.5 ± 3.8	86.0 ± 3.1	**86.5 ± 3.8**	23.8 ± 4.5
	10	83.9 ± 3.5	84.6 ± 3.8	83.9 ± 3.5	**85.0 ± 3.8**	23.8 ± 4.5
	20	82.0 ± 3.5	82.3 ± 4.2	83.1 ± 3.4	**84.0 ± 3.8**	23.8 ± 4.5
	35	0.0 ± 0.0	0.0 ± 0.0	81.8 ± 3.7	**82.0 ± 4.4**	23.8 ± 4.5
	50	0.0 ± 0.0	0.0 ± 0.0	**76.8 ± 5.1**	75.8 ± 4.9	23.8 ± 4.5
1501	1	**85.2 ± 3.9**	**85.2 ± 4.2**	**85.2 ± 3.9**	**85.2 ± 4.2**	23.8 ± 4.5
	300	75.8 ± 4.3	77.6 ± 4.2	76.8 ± 4.2	**79.4 ± 4.2**	23.4 ± 4.6
	600	54.3 ± 4.7	55.1 ± 5.4	71.4 ± 3.7	**72.2 ± 5.1**	22.9 ± 4.3
	1000	0.0 ± 0.0	0.0 ± 0.0	56.5 ± 5.1	**57.4 ± 4.6**	22.0 ± 4.6
	1400	0.0 ± 0.0	0.0 ± 0.0	**22.9 ± 2.6**	22.5 ± 3.2	21.8 ± 4.8
3001	1	**86.7 ± 2.7**	84.6 ± 2.9	**86.7 ± 2.7**	84.6 ± 2.9	23.8 ± 4.5
	600	67.7 ± 2.7	66.9 ± 4.0	68.1 ± 2.9	**71.5 ± 4.0**	22.9 ± 4.3
	1200	25.7 ± 5.8	25.8 ± 4.9	55.0 ± 4.2	**56.2 ± 3.8**	21.9 ± 4.7
	1800	0.0 ± 0.0	0.0 ± 0.0	**35.8 ± 4.8**	34.7 ± 4.0	21.5 ± 4.7
	2400	0.0 ± 0.0	0.0 ± 0.0	9.3 ± 3.0	9.9 ± 2.5	**20.5 ± 4.9**

TABLE XI: **Life Full Results**: Certified accuracy mean and standard deviation for the Life [62] dataset. Each ensemble submodel was trained on $\frac{1}{q}$-th of the training set with three q values tested per dataset, while kNN-CR was always trained on the whole training set (i.e., $q = 1$). The certified accuracy results of five robustness values (R) are reported per q value. Also reported as a baseline is the uncertified accuracy ($R = 0$) when training a single model on all of training set $\mathcal{S}$ ($q = 1$). Results are averaged across 10 trials per method, with each R's best mean certified accuracy in bold.

q	R	PCR	OCR	W-PCR	W-OCR	kNN-CR
1	0	92.7 ± 3.1	92.7 ± 3.1	92.7 ± 3.1	92.7 ± 3.1	34.6 ± 3.1
25	1	77.7 ± 4.4	**80.2 ± 4.0**	77.7 ± 4.4	78.3 ± 5.2	34.6 ± 3.1
	5	69.3 ± 4.9	**71.5 ± 4.7**	69.3 ± 4.9	71.4 ± 4.7	33.8 ± 2.8
	10	43.3 ± 5.6	54.2 ± 6.0	47.3 ± 5.9	**60.8 ± 4.9**	32.9 ± 2.8
	15	0.0 ± 0.0	0.0 ± 0.0	23.8 ± 4.0	**33.1 ± 3.2**	32.1 ± 2.9
	20	0.0 ± 0.0	0.0 ± 0.0	9.5 ± 2.9	11.4 ± 3.2	**31.1 ± 2.3**
101	1	71.1 ± 3.8	**71.5 ± 4.3**	71.1 ± 3.8	70.8 ± 4.1	34.6 ± 3.1
	10	58.9 ± 4.3	**61.8 ± 5.1**	58.9 ± 4.3	**61.8 ± 5.1**	32.9 ± 2.8
	20	40.5 ± 5.8	43.9 ± 4.3	40.5 ± 5.8	**45.4 ± 4.7**	31.1 ± 2.3
	30	20.7 ± 3.8	22.8 ± 4.4	20.7 ± 3.8	26.4 ± 3.9	**28.5 ± 2.5**
	40	4.4 ± 2.4	3.5 ± 1.4	4.6 ± 2.5	10.1 ± 2.7	**26.9 ± 2.4**
201	1	62.9 ± 4.1	**66.3 ± 3.0**	62.9 ± 4.1	65.7 ± 2.4	34.6 ± 3.1
	30	46.6 ± 3.8	49.0 ± 2.8	46.6 ± 3.8	**52.9 ± 3.0**	28.5 ± 2.5
	60	23.3 ± 2.7	24.6 ± 4.1	24.4 ± 2.6	**34.4 ± 3.9**	23.4 ± 2.3
	90	0.1 ± 0.3	0.6 ± 0.5	12.8 ± 2.9	18.0 ± 3.6	**18.1 ± 2.1**
	120	0.0 ± 0.0	0.0 ± 0.0	4.1 ± 1.6	4.4 ± 2.2	**8.5 ± 1.4**

TABLE XII: **Spambase Full Results**: Certified accuracy mean and standard deviation for the Spambase [63] dataset. Each ensemble submodel was trained on $\frac{1}{q}$-th of the training set with three q values tested per dataset, while kNN-CR was always trained on the whole training set (i.e., $q = 1$). The certified accuracy results of five robustness values (R) are reported per q value. Also reported as a baseline is the uncertified accuracy ($R = 0$) when training a single model on all of training set $\mathcal{S}$ ($q = 1$). Results are averaged across 10 trials per method, with each R's best mean certified accuracy in bold.

q	R	PCR	OCR	W-PCR	W-OCR	kNN-CR
1	0	87.5 ± 2.9	87.5 ± 2.9	87.5 ± 2.9	87.5 ± 2.9	64.0 ± 4.3
25	1	**87.6 ± 3.5**	87.1 ± 3.8	**87.6 ± 3.5**	85.8 ± 3.6	64.0 ± 4.3
	5	80.3 ± 3.8	81.0 ± 3.4	81.1 ± 3.6	**83.5 ± 3.7**	63.6 ± 4.1
	10	57.3 ± 4.5	65.0 ± 3.1	73.2 ± 3.8	**76.4 ± 3.8**	63.4 ± 4.3
	15	0.0 ± 0.0	0.0 ± 0.0	61.5 ± 4.8	63.1 ± 2.4	**63.2 ± 4.4**
	20	0.0 ± 0.0	0.0 ± 0.0	42.5 ± 4.4	38.7 ± 4.2	**63.0 ± 4.4**
151	1	**87.4 ± 2.9**	87.2 ± 2.2	**87.4 ± 2.9**	86.7 ± 2.6	64.0 ± 4.3
	25	69.1 ± 4.3	70.2 ± 5.5	69.1 ± 4.3	**75.7 ± 4.9**	63.0 ± 4.4
	50	22.8 ± 5.8	24.9 ± 4.0	35.4 ± 6.3	52.8 ± 4.0	**62.0 ± 4.8**
	75	0.0 ± 0.0	0.0 ± 0.0	14.8 ± 3.0	23.0 ± 3.9	**61.8 ± 4.7**
	100	0.0 ± 0.0	0.0 ± 0.0	3.4 ± 2.2	5.2 ± 2.4	**61.3 ± 4.2**
301	1	83.1 ± 2.8	**86.2 ± 3.3**	83.1 ± 2.8	86.0 ± 3.2	64.0 ± 4.3
	45	65.1 ± 4.7	68.6 ± 3.9	65.1 ± 4.7	**72.1 ± 3.9**	62.3 ± 4.3
	90	30.4 ± 3.5	34.6 ± 4.9	33.6 ± 3.2	53.7 ± 2.7	**61.7 ± 4.6**
	135	0.5 ± 0.7	0.1 ± 0.3	23.7 ± 3.5	31.1 ± 4.6	**60.2 ± 4.2**
	180	0.0 ± 0.0	0.0 ± 0.0	7.2 ± 2.5	11.3 ± 2.5	**58.3 ± 4.3**

B. kNN-CR *Full Certified Accuracy Plots*

To improve readability, Fig. 7 does not show kNN-CR's full certified accuracy trend. Instead, Fig. 8 below plots kNN-CR's full mean certified accuracy against that of W-OCR (using each dataset's maximum q value) for each of Sec. X's six datasets. Fig. 8 also visualizes the variance of each method by showing one standard deviation of the certified accuracy as a shaded region around the mean line. In summary, while W-OCR certifies more instances (i.e., has larger peak certified accuracy), its maximum certified robustness R is (significantly) smaller than that of kNN-CR.

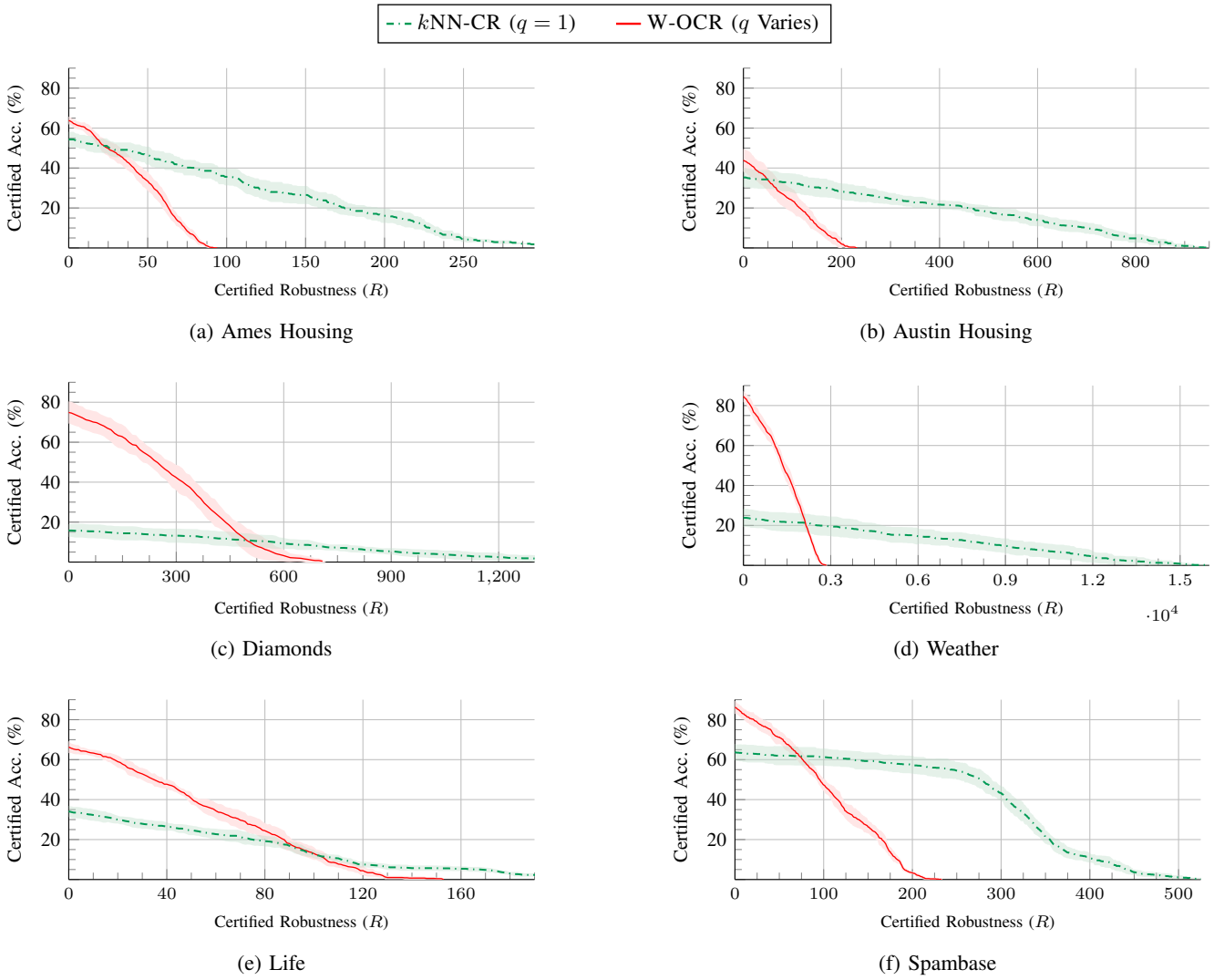


Fig. 8: **kNN-CR vs. W-OCR Certified Accuracy**: Full plots of the mean certified accuracy for Sec. X's six datasets. The shaded regions visualize one standard deviation of the certified accuracy for each R value. W-OCR's q value for each dataset is in Table XIII.

TABLE XIII: **W-OCR q Values**: As detailed in Sec. X-A, ensemble submodels were trained on $\frac{1}{q}$-th of the training data where q varies by dataset. Below are the W-OCR q values used in Fig. 8.

Dataset	Ames	Austin	Diamonds	Weather	Life	Spambase
q	251	701	1,001	3,001	201	301

C. Alternative ξ Evaluation

For each dataset in Sec. X, we evaluate a single threshold value ξ. This section explores how ξ affects our certified regressors' performance on four regression datasets.[16] Specifically, Fig. 9 considers our two best-performing methods, W-OCR and kNN-CR. Like in Sec X, kNN-CR uses $q = 1$. For W-OCR, we report results for each dataset's two largest q values. Note that in Table I, Ames and Diamonds use the same (default) threshold $\xi = 15\% \cdot y$ while both Life and Weather use (default) value $\xi = 3$. In Fig. 9 below, each dataset pair considers the same alternate ξ values.

ξ's exact effect varies across datasets, but generally, certified accuracy increases roughly linearly with ξ until the accuracy saturates.

[16]For binary classification (e.g., Spambase [63]), alternate ξ values do not apply.

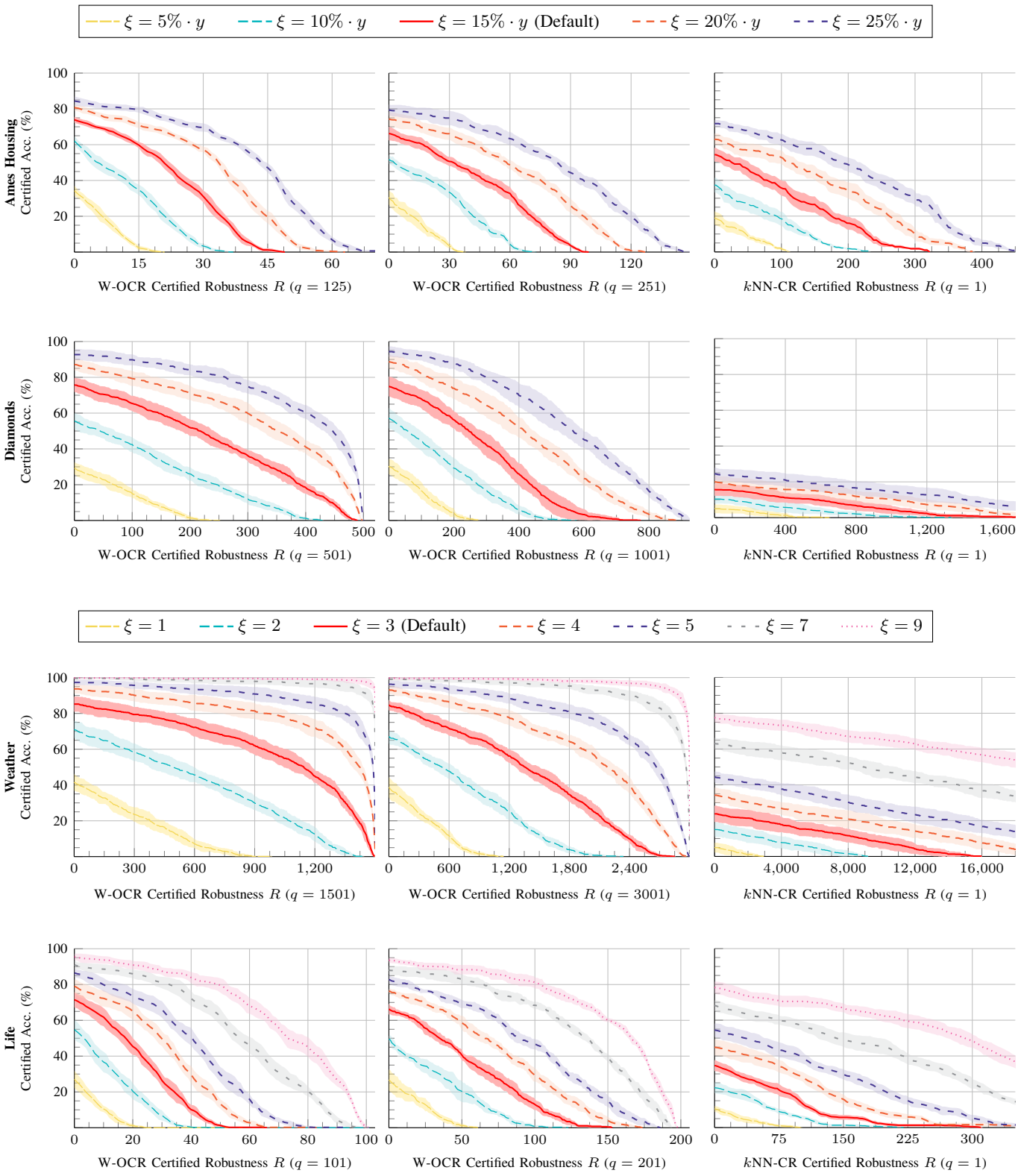


Fig. 9: **Alternative ξ Values**: Mean certified accuracy of our two best performing methods, W-OCR and kNN-CR, across alternative error thresholds ξ. For W-OCR, we consider each dataset's two largest q values from Fig. 7. The "default" ξ value in each sublegend denotes the dataset's corresponding ξ value that is evaluated in Sec. X (Tab. I). Each line's shaded region visualizes one standard deviation around the mean certified accuracy.

D. Model Training Times

This section summarizes the time needed to train our primary certified regressors. All experiments were performed using a single core of a fourteen-core Intel E5-2690v4 CPU with 12GB of 2400MHz DDR4 RAM. As is standard for kNN models, kNN-CR's training time is essentially zero since "training" simply entails instance memorization, which is nearly instantaneous.

Table XIV details the training times of our two partitioned regressors: PCR which follows the unit-cost assumption (i.e., $\forall_t\, r_t = 1$) and its weighted extension W-PCR. The training times are broken down into the time required to train a single submodel as well as the average time to train the whole ensemble. As expected, W-PCR takes much longer to train than PCR since the former requires training $\mathcal{O}(n+T)$ models (for $r_{\max} = 2$) while the latter requires only $\mathcal{O}(T)$ models.

As q increases, each submodel is trained on fewer data. It might be assumed that submodel training time is inversely related to q. However, Table XIV shows that this is not necessarily the case since different hyperparameter settings can drastically affect how long a model takes to train, even after accounting for q.

Note also that all ensemble submodels can be trained fully in parallel. Hence, by simply parallelizing submodel training, PCR's total training time can be sped up by a factor of about T while W-PCR's total training time can be sped up by about a factor of $(n+T)$. Moreover, while training an ensemble (partitioned or overlapping) is often more computationally expensive than training just one model on the whole training set, this additional training time is amortized across all test predictions that need to be certified.

Lastly, recall from Sec. X-A's evaluation that our overlapping certified ensembles, OCR and W-OCR, have $T = qd$ submodels, where d is the training set block spread degree. Therefore, the time to train OCR and W-OCR ensembles is about d times larger than that of PCR and W-PCR, respectively. See Table I for each dataset's d value.

TABLE XIV: **(Weighted) Partitioned Certified Regression Model Training Times**: Mean and standard deviation PCR and W-PCR model training times (in seconds) for six datasets in Sec. X. Below each dataset name is the corresponding submodel architecture that was used – either ridge regression or XGBoost. All experiments were performed on a single CPU core.

Dataset (Model Type)	q	PCR		W-PCR	
		Submodel	Total	Submodel	Total
Ames (XGBoost)	25	2.5 ± 0.50	63.1	213.6 ± 28.3	5340
	125	3.1 ± 0.33	384	27.5 ± 9.8	3437
	251	1.0 ± 0.13	261	3.9 ± 1.7	984
Austin (XGBoost)	51	2.2 ± 0.19	111	502.3 ± 109.3	25,617
	301	0.37 ± 0.06	184	25.5 ± 8.9	7,688
	701	0.89 ± 0.11	626	12.5 ± 8.1	8,786
Diamonds (Ridge)	151	0.01 ± 0.00	0.7	2.1 ± 0.2	322
	501	0.01 ± 0.00	1.9	0.6 ± 0.1	293
	1,001	0.01 ± 0.00	4.2	0.3 ± 0.1	299
Weather (Ridge)	51	0.05 ± 0.02	2.4	298.7 ± 109.7	15,234
	1,501	0.01 ± 0.00	12.6	2.5 ± 0.2	3,776
	3,001	0.02 ± 0.00	66.0	4.7 ± 10.6	14,051
Life (XGBoost)	25	2.2 ± 0.18	56.2	203.8 ± 34.9	5,095
	101	1.8 ± 0.31	185	33.5 ± 12.6	3,388
	201	8.1 ± 2.6	1,624	85.5 ± 54.6	17,184
Spambase (Ridge)	25	0.03 ± 0.01	0.8	4.4 ± 1.2	109
	151	0.01 ± 0.00	1.7	0.5 ± 0.3	69
	301	0.01 ± 0.00	0.9	0.1 ± 0.0	19

E. Overlapping Regression ILP Execution Time

Recall from Sec. VIII that our two overlapping certified regressors, OCR and W-OCR, use an integer linear program to bound certified robustness R. Under OCR's unit-cost structure where $\forall_t\, r_t \leq 1$, Fig. 6 is actually a *binary integer program* since $\forall_j\, \omega^{(j)} \in \{0,1\}$. In contrast, with W-OCR's weighted costs, Fig. 6 is a true integer linear program. While both OCR and W-OCR have the same asymptotic complexity, we observed that LP solvers generally solve programs with only binary variables faster than integral ones.

Sec. X's evaluation implemented Fig. 6's LP in Gurobi [54]. All experiments were performed on a single core of a fourteen-core Intel E5-2690v4 CPU with 12GB of RAM. Fig. 10 shows the ILP execution time distribution of OCR and W-OCR for three datasets from Sec. X. For each dataset, we report the ILP's execution time for the two largest q values. Each histogram visualizes at least 1600 trials per method.

In summary, ILP execution time is bimodal with run times clustered around 0 seconds and 1200 seconds (i.e., the ILP's time limit). OCR's ILP is generally faster on average than W-OCR (observe that OCR times out less frequently, if at all).

Note also that the ILP's execution time distribution does not change significantly across each dataset's two q values.

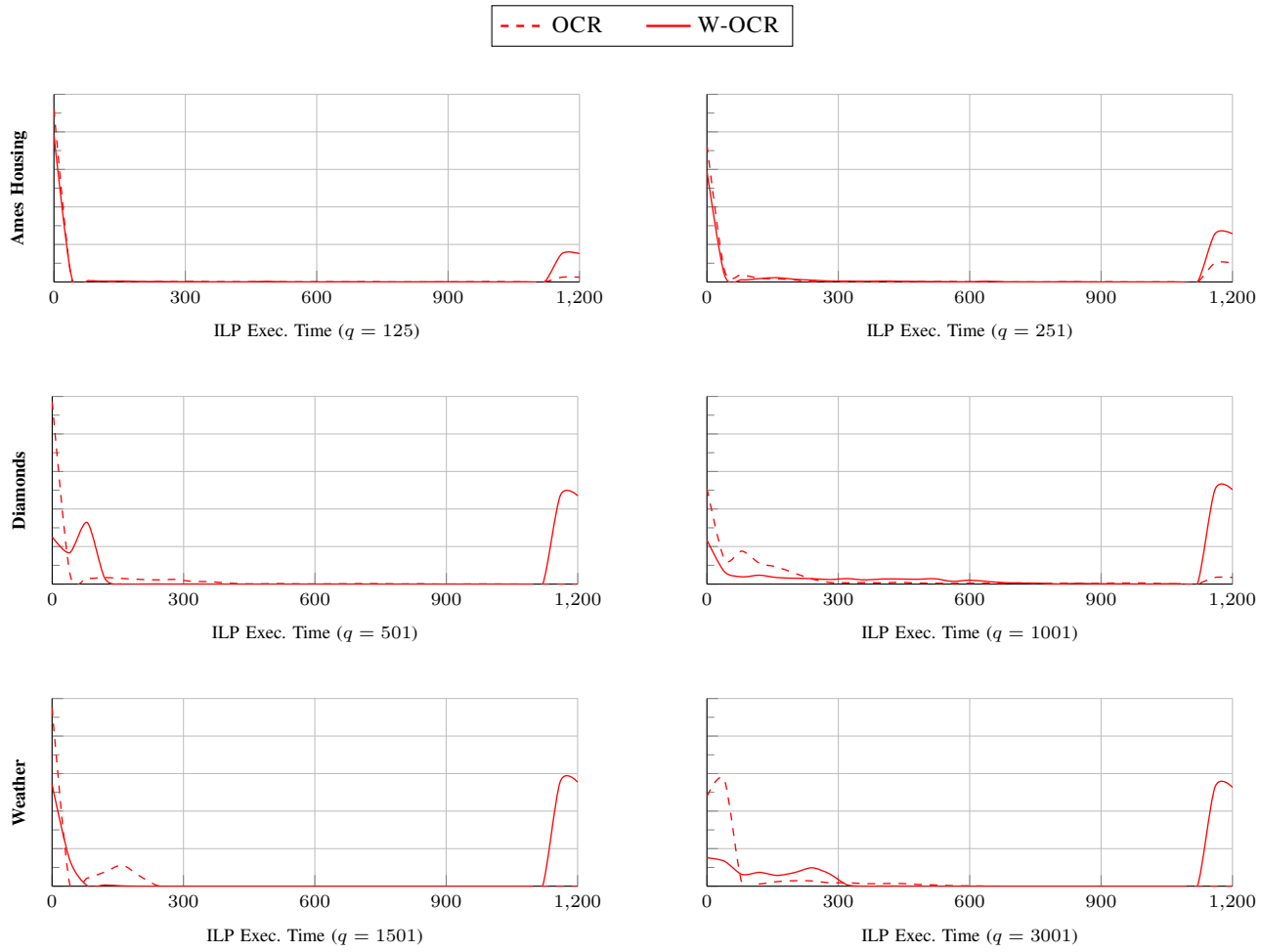


Fig. 10: **Gurobi ILP Execution Time**: Histogram of the Gurobi ILP execution time (in seconds) for OCR and W-OCR for three datasets from Sec. X. ILP execution times for each dataset's two largest q values are plotted, with each of dataset plots sharing the y-axis scales. The ILP was implemented in Gurobi with a fixed time limit of 1200 seconds.

F. Effect of Median as the Model Decision Function

This section evaluates how the choice of decision function affects the accuracy of standard (i.e., non-robust) regression. All experiments above exclusively used median as the decision function. Below, we compare median's performance to the more traditional decision function, mean. To be clear, this section's experiments exclusively consider non-robust model predictions. This mini ablation study simply examines the decision function's effect on baseline (uncertified) prediction accuracy.

Table XV compares the performance of mean and median on Sec. X's five regression datasets. These experiments used the same threshold (ξ) values as Sec. X (see Tab. I). Tab. XV includes results for two different model architectures: kNN-based regression and a regression ensemble that follows PCR's base architecture. For each dataset, the ensemble regressor is evaluated on the same three q values used in Sec. X.

In summary, median and mean decision functions have comparable performance. Median had better average accuracy than mean on 11 of 20 evaluation setups. Median and mean had equivalent average performance on four setups. Mean outperformed median on the remaining five setups.

kNN regression was the method most affected by the choice of decision function. In particular for the Weather and Life datasets, mean outperformed median by 8.6 and 2.3 percentage points, respectively. Only one other evaluation setup (Austin with $q = 701$) saw a similarly large performance difference (3.8pp).

TABLE XV: **Effect of Median vs. Mean as the Decision Function**: Comparison of the model accuracy mean and standard deviation for two different decision functions. For each of Sec. X's five regression datasets, we evaluate the decision function's effect on both kNN and ensemble (PCR) learners. Each dataset's PCR non-robust accuracy is reported for three different q values in line with Sec. X's evaluation. For each experimental setup, the best performing method (in terms of average accuracy) is shown in bold. In summary, median and mean decision functions have comparable baseline (uncertified) prediction accuracy. However, median is critical to achieve certified robustness guarantees.

Dataset	Model	q	Decision Function	
			Median	Mean
Ames	kNN	1	**53.4 ± 5.4**	**53.4 ± 5.4**
	PCR	25	83.8 ± 2.7	**84.6 ± 2.6**
		125	**71.2 ± 4.0**	71.0 ± 3.9
		251	**63.4 ± 3.5**	62.7 ± 2.2
Austin	kNN	1	**32.9 ± 3.8**	**32.9 ± 3.8**
	PCR	51	61.4 ± 4.9	**61.8 ± 4.5**
		301	51.6 ± 4.0	**51.8 ± 3.9**
		701	**44.6 ± 4.8**	40.8 ± 5.5
Diamonds	kNN	1	**16.4 ± 2.1**	**16.4 ± 2.1**
	PCR	151	**74.9 ± 3.8**	73.6 ± 4.4
		501	**77.7 ± 4.4**	76.6 ± 5.4
		1001	**75.2 ± 4.1**	73.2 ± 4.4
Weather	kNN	1	24.6 ± 4.5	**33.2 ± 5.4**
	PCR	51	**86.3 ± 3.2**	86.0 ± 3.2
		1501	**85.2 ± 3.9**	85.0 ± 3.8
		3001	**86.7 ± 2.7**	**86.7 ± 2.7**
Life	kNN	1	35.7 ± 3.2	**38.0 ± 3.5**
	PCR	25	**80.1 ± 4.0**	79.9 ± 3.7
		101	**72.0 ± 3.2**	71.5 ± 3.7
		201	**63.3 ± 4.1**	61.8 ± 3.3