MetaMoE: Formal Verification of Compositional Robustness and Scalability of Mixture-of-Experts Architecture

Quang Pham, Ben Wooding, Luke Nam, Samuel Sasaki, and Taylor T. Johnson

Institute for Software Integrated Systems, Vanderbilt University, Nashville, TN, USA
{quang.m.pham, ben.wooding, luke.nam, samuel.sasaki, taylor.johnson}@vanderbilt.edu

Abstract. Mixture-of-experts (MoE) architectures offer modularity and scalability, yet their robustness, and the practicality of certifying that robustness, in heterogeneous settings are not well understood. This work presents MetaMoE, a heterogeneous MoE framework designed for compositional formal verification. In MetaMoE, a neural network, called a router, classifies an input image's domain and routes it to the corresponding domain expert neural network for fine-grained classification; we study how robustness propagates compositionally across these router and experts and establish when system-level verification can be derived from component-level verification. In homogeneous MoE, all experts share the same task, so a misroute may still preserve correctness if the receiving expert classifies the input correctly; in heterogeneous MoE, experts operate on disjoint class spaces, making any misroute catastrophic. We prove that when the router maintains expert selection under hard routing (k=1) within a perturbation limit, and the selected expert also maintains its classification within that perturbation limit, the end-to-end system robustness is compositional. This enables scalability as the computational power needed for verification and re-verification does not snowball as the system expands. We further complement the formal verification results with empirical experiments, which support the same robustness trends observed under certification. Experiments across 8 MoE configurations and 3 perturbation budgets show that robustly trained (RT) experts improve adversarial accuracy by up to 13.1% over non-robustly trained (NRT) ones and are a prerequisite for formal certifiability – NRT models on complex domains are completely unverifiable – while router training paradigm has negligible impact ($< 0.1\%$), and verified routers achieve 100% certified robustness accuracy (RoCRA) at practical perturbation bounds. These results outline a principled path toward scalable, verifiable modular AI.

Keywords: Mixture-of-experts · Adversarial Robustness · Formal Verification · Compositional Verification · Neural Network Verification · Safety-Critical AI

1 Introduction

Deep neural networks (DNNs) achieve high accuracy on image classification tasks but remain vulnerable to small adversarial perturbations [10,22]. This unreliability is hard to justify in safety-critical vision applications such as autonomous driving, traffic-sign recognition, and medical imaging. In parallel, Mixture-of-experts (MoE) architectures [12,33], originally proposed in the early 1990s and recently revitalized for large-scale deep learning, offer a powerful paradigm for scaling models through modular computation and conditional activation. The core idea is to replace a single monolithic neural network with a collection of smaller, specialized sub-networks called *experts*, each is trained to handle a specific subset of input space, together with a neural network *router* that examines each input and decides which expert(s) should process it. Since only one or a few experts are activated per input, MoE systems' computation and inference costs increase slower than system capacity. While recent work has popularized MoE primarily within large language models [33,7], the paradigm is equally viable for *vision* systems [31] where distinct visual domains (*e.g.*, traffic signs [35], handwritten digits [20], *etc.*) benefit from dedicated expert networks. Yet the robustness properties of such heterogeneous vision MoE systems, where domain-specific experts must cooperate under a shared router, remain poorly understood.

Formal verification tools such as α–β-CROWN [41,38] can certify provable robustness for individual neural networks, yet verifying a full MoE system remains challenging because the verifier must reason jointly about the router, every expert, and their interactions. This raises a natural question: can the verified robustness of individual experts and the router be combined to certify the full system? An affirmative answer would allow heterogeneous MoE systems to be trained, extended with new experts, and certified modularly, avoiding the need to re-verify the entire system whenever a single component changes.

This paper introduces MetaMoE (Figure 1), a verification-aware heterogeneous MoE architecture in which each input carries two labels: a fine-grained *class* (*e.g.*, speed-limit sign) and a coarse *meta-class* identifying the domain (*e.g.*, German traffic signs vs. handwritten digits). The router learns to predict the meta-class and dispatches the input to the corresponding domain expert, which then predicts the fine-grained class. The framework combines verifiable experts with a verifiable routing network, intermixing adversarially and non-adversarially trained components, and is complemented by empirical adversarial experiments under Projected Gradient Descent (PGD) attacks [22] that test whether the same robustness trends hold across model variants. Under hard routing, exactly one expert processes each input, so the system decomposes naturally into two independent verification tasks: does the router select the right expert?, and does that expert classify correctly? If both hold, the system is correct. This decomposition leads to three contributions, each building on the previous:

1. **Compositional Robustness (Theorem 1).** We prove an assume-guarantee rule for hard-routed MoE with disjoint expert class spaces: the system is robust at an input if and only if (i) the router maintains invariant expert

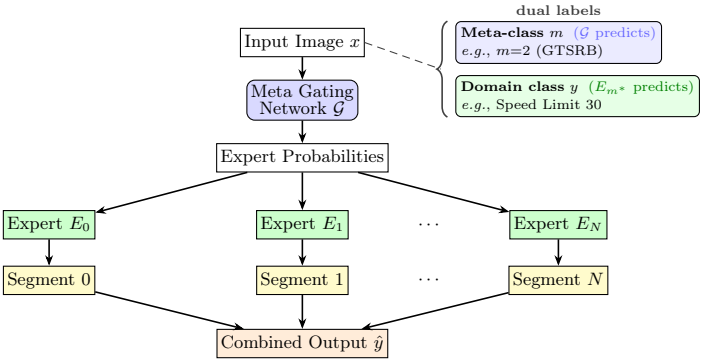

Fig. 1. MetaMoE architecture. Each input carries two labels: a coarse *meta-class* m identifying the domain and a fine-grained *domain class* y. The meta gating network $\mathcal{G}$ predicts m to route the input to the corresponding expert E_{m^*}, which then predicts y. Under hard routing (k=1), the top-scoring expert's output populates the corresponding segment of the unified output vector $\hat{y} \in \mathbb{R}^{C_{\text{total}}}$ (Eq. 1), with all other segments zeroed. Experts are frozen after independent training; only $\mathcal{G}$ is trained during MoE composition.

selection within the ℓ_∞ perturbation ball and (ii) the selected expert is robust within that same ball. The biconditional guarantees that no interaction effects between router and expert can invalidate component-level certificates, reducing system-level verification to independent tasks whose cost is fixed regardless of how many experts the system contains.

2. **Verification methodology.** We leverage the theorem in a practical compositional verification pipeline using α–β-CROWN, a state-of-the-art neural network verifier. Because the theorem guarantees soundness of independent component verification, we extract the router as a standalone network and verify it separately from the experts. We describe the architecture and export decisions (BatchNorm elimination, raw logit output, and property negation for counterexample search) that make each component amenable to bound propagation. This reduces the verification target from the full system (∼231K parameters) to the largest single component (∼96K), with the reduction ratio scaling linearly with the number of experts.
3. **Empirical case study.** We support the formal findings with an empirical study across 8 MoE configurations (40 training runs), 3 perturbation budgets ($\varepsilon = 2/255, 4/255, 8/255$), and 4 expert/router training combinations on CIFAR-10 [19], MNIST [20], GTSRB [35], and PTSD [24]. The study reveals three practical findings: (i) for visually dissimilar domains, expert robustness dominates system behavior while router adversarial training has negligible impact ($< 0.1\%$ accuracy difference), meaning the router can be trained without costly adversarial augmentation; (ii) within each domain and architecture, empirical adversarial accuracy (AA) supports the same model ordering observed under certified robustness (ExCRA); and (iii) robust

training is necessary to obtain nontrivial certified robustness in our setting, as non-robustly trained models on complex domains are completely unverifiable.

2 Related Work

Sparse gating [33] made MoE [12] practical at scale, and subsequent work extended the paradigm to vision [31], showed that simplified top-1 routing scales MoE to trillion-parameter models with reduced communication cost [9], and scaled sparse routing to hundreds of billions of parameters [6,13,7]. Complementary advances in catastrophic-forgetting mitigation [30], gating evolution from unsupervised [14] to supervised [31] schemes, and modular deep learning [26] have established MoE as a mature paradigm for composable model design.

Yet this scalability work largely ignores a critical question: *are MoE systems robust, and can that robustness be formally certified?* The few studies that examine MoE robustness each address a different facet of this question without resolving it holistically. Puigcerver et al. [27] showed that sparse MoE Vision Transformers can be more robust than dense counterparts at the same computation cost, but their formal guarantees are derived only for simplified settings with smooth MoEs or linear experts under fixed routing and structured data-separation assumptions. Zhang et al. [44] studied model-level MoE ensembles of identical experts and found that *within* such systems, the expert networks are the primary vulnerability (far more susceptible to adversarial attack than the router), a complementary rather than contradictory finding since it characterizes where vulnerability concentrates rather than whether MoE improves overall robustness. Pavlitska et al. [25] confirmed that sparse MoE layers inside CNNs improve adversarial robustness when combined with adversarial training, though MoE structure alone confers negligible benefit. All three studies examine *homogeneous* MoE (same experts, same dataset); none consider *heterogeneous* MoE with disjoint expert domains, nor do they analyze how robustness propagates through the router-expert composition or connect empirical resilience to provable certificates.

Meanwhile, neural network verification has matured as an independent line of research. Katz et al. [15] introduced Reluplex and established the NP-completeness of ReLU network verification, and Alsmann and Lange [1] recently extended complexity analysis to state space models, showing that satisfiability is undecidable in the general case and NP-complete under bounded context. These hardness results motivate scalable approximations: α–β-CROWN [41,38] provides state-of-the-art complete verification via bound propagation (current winner of VNN-COMP [16]), NNV [37] offers reachability-based verification for cyber-physical systems, and GRENA [45] scales abstract refinement to CNNs via GPU-accelerated unconstrained optimization. Certified training methods such as CROWN-IBP [43] and randomized smoothing [5] produce models with built-in provable guarantees, while the robustness margin [17] characterizes robustness independently of a fixed ε. PGD-based adversarial training [22] provides complementary empirical robustness, and probabilistic verification [4] bridges the gap between formal and statistical guarantees. However, all of these tools and techniques target *mono-*

lithic networks; they cannot directly certify a system whose behavior depends on dynamic routing among multiple sub-networks.

One promising solution is compositional verification, long established in classical systems through assume-guarantee reasoning [11]. Zhou and Tripakis [46] recently adapted compositional inductive invariants to neural network controlled systems, demonstrating that modular proofs can scale where monolithic analysis cannot. Watson et al. [39] decompose verification of autonomous perception systems into environment-specific scenarios with composed probabilistic guarantees. However, both frameworks address feedback control loops or perception pipelines, not the conditional-activation structure of MoE, where a router's decision determines which expert's guarantees apply.

These three lines of work (MoE scalability, adversarial robustness, and formal verification) have developed largely in isolation. MoE research scales models without certifying them; robustness studies attack or defend MoE empirically without formal guarantees; and verification certifies monolithic networks without addressing compositional, dynamically-routed architectures. This paper bridges all three by asking: *Can compositional robustness in heterogeneous MoE systems be formally characterized, formally certified, and empirically supported?* We answer affirmatively by proving that router invariance and expert robustness compose to yield system-level guarantees, and by showing that empirical experiments support the same robustness trends observed under certification in our study.

3 Problem Formulation

We define the following notation used throughout this paper. Let $x \in \mathbb{R}^{C\times H\times W}$ denote a normalized input image, where $C = 3$ is the number of color channels (RGB), H is the image height, and W is the image width (in our experiments, $H = W = 32$ pixels). Let N denote the total number of domain-specific experts in the system, and let C_i denote the number of classes in dataset i (*e.g.*, $C_i = 10$ for CIFAR-10, $C_i = 43$ for GTSRB). The total number of classes across all datasets is $C_{\text{total}} = \sum_{i=1}^{N} C_i$. Each input carries two labels: a ground-truth *meta-class* $m \in \{0, \ldots, N-1\}$ identifying which dataset (and hence which expert) the input belongs to, and a fine-grained *class label* $y \in \{1, \ldots, C_{\text{total}}\}$.

A heterogeneous MoE system consists of two types of components:

1. **Experts:** Each expert E_i is a neural network specialized for dataset i. It maps an input image to a vector of class logits (unnormalized scores) for its domain: $E_i : \mathbb{R}^{C\times H\times W} \to \mathbb{R}^{C_i}$, producing $y_i = E_i(x)$ where $y_i \in \mathbb{R}^{C_i}$ is the logit vector for domain i.
2. **Router:** The router (also called gating network) $\mathcal{G}$ determines which expert should handle a given input. It maps the input image to N routing logits: $\mathcal{G} : \mathbb{R}^{C\times H\times W} \to \mathbb{R}^{N}$, producing $g = \mathcal{G}(x)$ where g_i is the routing score for expert i.

Under hard routing (top-1 selection), the router selects a single expert $m^* = \arg\max_i g_i$, i.e., the expert with the highest routing score. For top-k routing

(where k experts are activated simultaneously), let $\mathcal{S} = \{s_1, \ldots, s_k\} \subset \{1, \ldots, N\}$ denote the indices of the k experts with the highest routing scores. Each selected expert s_j receives a normalized weight:

$$w_j = \frac{g_{s_j}}{\sum_{\ell=1}^{k} g_{s_\ell}}, \quad j = 1, \ldots, k$$

representing its contribution to the final output. To combine outputs from experts with different class spaces, we define *class offsets* $o_i = \sum_{m<i} C_m$ so that expert i's classes occupy positions o_i through $o_i + C_i - 1$ in a unified output vector $\hat{y} \in \mathbb{R}^{C_{\text{total}}}$. For each output index c, let $\mathcal{A}(c) = \{\, j \in \{1, \ldots, k\} : c \in \{o_{s_j}, \ldots, o_{s_j} + C_{s_j} - 1\} \,\}$ denote the subset of activated experts whose class range contains c. The unified MoE output $\hat{y}$ (distinct from the target label y) is:

$$\hat{y}_c = \begin{cases} \displaystyle\sum_{j \in \mathcal{A}(c)} \frac{w_j}{\sum_{\ell \in \mathcal{A}(c)} w_\ell}\, y_{s_j}[c - o_{s_j}], & \text{if } \mathcal{A}(c) \neq \emptyset \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

where $y_{s_j}[c - o_{s_j}]$ is expert s_j's logit at local class index $c - o_{s_j}$, and the per-index renormalization makes contributions a convex combination of expert logits when multiple activated experts share that class. In the disjoint-class hard-routing regime verified here, $|\mathcal{A}(c)| \leq 1$ and (1) collapses to $\hat{y}_c = y_{s_{j^*}}[c - o_{s_{j^*}}]$. Proving the general form for the soft and top-k extensions is left as future work, see Section 6.6.

3.1 Adversarial Threat Model and Robustness

We consider an adversary that applies a small perturbation $\delta \in \mathbb{R}^{C \times H \times W}$ to the input, bounded under the ℓ_∞ norm: $\|\delta\|_\infty \leq \varepsilon$, where $\varepsilon > 0$ is the perturbation budget (*e.g.*, $\varepsilon = 8/255 \approx 0.031$ in pixel intensity). This produces adversarial examples $x' = x + \delta$ lying within the ℓ_∞ ball $B_\varepsilon(x) = \{x' : \|x' - x\|_\infty \leq \varepsilon\}$, i.e., all images within ε of x in every pixel channel. We keep the verification-literature term "ℓ_∞ ball," but $B_\varepsilon(x)$ is geometrically an axis-aligned hypercube of side 2ε; our certified guarantees hold over this cube and not over any wider neighborhood of x. Robustness requires that the model's prediction remains correct for *all* perturbations within this ball:

- **Expert E_i robustness:** Given that the router has already selected expert i, the expert itself must still predict the correct class even when the input is slightly perturbed. Formally, the expert's predicted class is unchanged: $\arg\max_c\, y_i(x')[c] = y$ for all $x' \in B_\varepsilon(x)$, where $y_i(x') = E_i(x')$ is the logit vector of expert i on the perturbed input. In other words, no small perturbation can cause the expert to misclassify the input within its own task domain.
- **Router $\mathcal{G}$ robustness (invariance):** The router must consistently delegate the input to the same expert regardless of small perturbations, ensuring

that an adversary cannot trick the system into activating the wrong expert. Formally, $m^*(x') = m^*(x)$ for all $x' \in B_\varepsilon(x)$, meaning the router's expert selection is invariant within the ε-ball. Equivalently, the selected expert's routing logit remains strictly dominant over all alternatives: $g_{m^*}(x') - g_j(x') > 0$ for all $j \neq m^*$ and all $x' \in B_\varepsilon(x)$.

- **MoE system robustness:** The end-to-end system – comprising both the router and the selected expert – must produce the correct final classification under perturbation. This is the strongest requirement, as it demands that neither the routing decision nor the expert's prediction is corrupted. Formally, $\arg\max_c \hat{y}_c(x') = y$ for all $x' \in B_\varepsilon(x)$, where $\hat{y}(x')$ is the unified MoE output (1).

We quantify verifiability with two *certified* metrics, each computed by a formal verifier over a finite test set D. For a single test image (x, y), the verifier returns a binary verdict: either it *proves* that the property holds for every perturbation in $B_\varepsilon(x)$, or it cannot (timeout or counterexample). The aggregate metric is the fraction of images in D that receive a positive verdict.

Expert Certified Robustness Accuracy (ExCRA). For an expert E_i, the per-image property is correct classification under all perturbations: $\arg\max_c E_{i,c}(x') = y$ for all $x' \in B_\varepsilon(x)$, where $E_{i,c}(\cdot)$ denotes the c-th logit of expert i's output. The aggregate metric is:

$$\mathrm{ExCRA}(\varepsilon) = \frac{1}{|D|} \sum_{(x,y)\in D} \mathbb{1}\big[\mathrm{Verify}(E_i, x, \varepsilon)\big] \tag{2}$$

Router Certified Robustness Accuracy (RoCRA). For the router $\mathcal{G}$, the per-image property is invariant expert selection: $\arg\max_i \mathcal{G}_i(x') = m^*$ for all $x' \in B_\varepsilon(x)$, where $\mathcal{G}_i(\cdot)$ denotes the i-th routing logit and $m^* = \arg\max_i \mathcal{G}_i(x)$. The aggregate metric is:

$$\mathrm{RoCRA}(\varepsilon) = \frac{1}{|D|} \sum_{(x,m^*)\in D} \mathbb{1}\big[\mathrm{Verify}(\mathcal{G}, x, \varepsilon)\big] \tag{3}$$

In both cases, $\mathbb{1}[\mathrm{Verify}(\cdot)]$ equals 1 only when the verifier produces a formal proof; timeouts and counterexamples both yield 0, so ExCRA and RoCRA lower-bound true robustness. Because these metrics aggregate binary formal verdicts over a test set, a reported value such as "ExCRA = 90%" means "the verifier formally certified 18 out of 20 test images."

Relation to constraint-based metrics. Casadio et al. [3] cast robustness as a verification property with three metrics: *constraint satisfaction* (per-input verdict), *constraint accuracy* (fraction satisfying it on the clean point alone), and *constraint security* (fraction for which it holds throughout the perturbation region). In their terms, our $\mathbb{1}[\mathrm{Verify}(\cdot)]$ is a constraint-satisfaction verdict, ExCRA and RoCRA are constraint-security aggregates over expert correctness and router invariance, and clean accuracy is constraint accuracy. Theorem 1 then reads as composing two constraint-security guarantees into a system-level one.

Adversarial Accuracy (AA). As an empirical complement to the certified metrics, we measure the fraction of test images that remain correctly classified under PGD attack [22]: $\mathrm{AA}(\varepsilon) = \frac{1}{|D|}\sum_{(x,y)\in D} \mathbb{1}[\arg\max_c f_c(x^{\mathrm{adv}}) = y]$, where $x^{\mathrm{adv}} = \arg\max_{x'\in B_\varepsilon(x)} \mathcal{L}(f(x'), y)$ is a PGD-generated adversarial example and f is the model under evaluation (expert or full MoE). AA upper-bounds true robustness since PGD is heuristic.

3.2 Problem Statement

Given N pretrained experts $\{E_i\}$, router $\mathcal{G}$, and L_∞ adversary with radius ε, we investigate:

1. **Compositional Robustness:** Does robust training of components yield provably robust MoE composition?
2. **Empirical Support for Formal Findings:** Do AA experiments support the same robustness trends and model ordering observed under ExCRA analysis?
3. **Scalability:** Can modular robustness achieve reduced retraining time, linear inference $O(k)$, and stable verification?

3.3 Compositional Robustness Theorem

Rather than verifying the MoE as a monolithic whole, we adopt assume-guarantee reasoning [11], in which each component *assumes* certain environmental conditions and *guarantees* a local property. In our setting, the router guarantees stable expert selection and each expert guarantees correct classification within its domain. We formalize this decomposition below through a definition, two supporting lemmas, and a composing theorem showing that these component-level guarantees are individually necessary and jointly sufficient for system robustness.

Definition 1 (Router Invariance). *Router $\mathcal{G}$ is* invariant *at input x within perturbation ball $B_\varepsilon(x)$ if, for every perturbed input $x' \in B_\varepsilon(x)$,*

$$\arg\max_i \mathcal{G}_i(x') = \arg\max_i \mathcal{G}_i(x).$$

That is, no perturbation within the ε-ball can cause the router to select a different expert.

Lemma 1 (Router Invariance Preserves Expert Selection). *Let $m^* = \arg\max_i \mathcal{G}_i(x)$ be the expert selected for clean input x. If $\mathcal{G}$ is invariant at x within $B_\varepsilon(x)$ (Definition 1), then $m^*(x') = m^*$ for all $x' \in B_\varepsilon(x)$, and the MoE output reduces to a fixed expert's output: $\hat{y}(x') = E_{m^*}(x')$ (up to class offsets) for all $x' \in B_\varepsilon(x)$.*

Proof. By Definition 1, $\arg\max_i \mathcal{G}_i(x') = m^*$ for all $x' \in B_\varepsilon(x)$. Under hard routing (k=1), only expert m^* is activated, so $\hat{y}_c(x') = E_{m^*}(x')[c - o_{m^*}]$ for $c \in \{o_{m^*}, \ldots, o_{m^*} + C_{m^*} - 1\}$ and $\hat{y}_c(x') = 0$ otherwise. The MoE output is therefore fully determined by E_{m^*}. □

Lemma 1 eliminates the routing dimension of the problem: under hard routing, an invariant router selects the same expert for every input in the perturbation ball, so the full MoE output depends only on that expert. The remaining question is whether that expert classifies correctly throughout the ball.

Lemma 2 (Expert Robustness Preserves Classification). *If expert E_{m^*} is robust at x within $B_\varepsilon(x)$, i.e.,*

$$\arg\max_c E_{m^*,c}(x') = y \quad \textit{for all } x' \in B_\varepsilon(x),$$

and the MoE output is determined solely by E_{m^*}, then $\arg\max_c \hat{y}_c(x') = y$ for all $x' \in B_\varepsilon(x)$.

Proof. Since the MoE output vector $\hat{y}(x')$ is zero outside the class range of E_{m^*} and equals $E_{m^*}(x')$ within that range (by Lemma 1), the argmax of $\hat{y}(x')$ coincides with the argmax of $E_{m^*}(x')$ (shifted by offset o_{m^*}). By assumption, this argmax equals y for all $x' \in B_\varepsilon(x)$. □

Theorem 1 (Compositional Robustness). *Consider a hard-routed MoE (k=1) with router $\mathcal{G}$ and experts $\{E_i\}$ having disjoint class spaces. Let $m^* = \arg\max_i \mathcal{G}_i(x)$ denote the selected expert for input x. The MoE system is robust at x within $B_\varepsilon(x)$ if and only if:*

(i) $\mathcal{G}$ is invariant at x within $B_\varepsilon(x)$ (Lemma 1), and
(ii) E_{m^*} is robust at x within $B_\varepsilon(x)$ (Lemma 2).

That is, for disjoint expert class spaces, where m^* is the router-selected expert:

$$\textit{MoE robust at } x \iff \underbrace{\textit{Router invariant at } x}_{\textit{same expert } m^* \textit{ selected}} \wedge \underbrace{\textit{Expert } E_{m^*} \textit{ robust at } x}_{\textit{correct class predicted}} \quad (4)$$

Proof. **(Sufficiency.)** Follows directly by composing Lemmas 1 and 2: router invariance ensures the same expert m^* is selected for all $x' \in B_\varepsilon(x)$, and expert robustness ensures that expert produces the correct output throughout.

(Necessity, under disjoint classes.) Suppose condition (i) fails: there exists $x' \in B_\varepsilon(x)$ with $m^*(x') = m' \neq m^*$. Since experts have disjoint class spaces, $E_{m'}$ outputs logits only over m''s classes. Because robustness requires correct classification on all inputs in $B_\varepsilon(x)$, including the clean input x itself, and the clean system correctly classifies x, the true label y must lie in m^*'s class range. Hence $\arg\max_c \hat{y}_c(x') \neq y$, and the MoE is not robust. Now suppose (i) holds but (ii) fails: there exists $x' \in B_\varepsilon(x)$ with $\arg\max_c E_{m^*,c}(x') \neq y$. By Lemma 1, $\hat{y}(x')$ is determined by $E_{m^*}(x')$, so $\arg\max_c \hat{y}_c(x') \neq y$, and again the MoE is not robust. □

The practical consequence of this assume-guarantee decomposition is *modular verification* (Figure 2): verify router invariance and expert robustness independently, then compose guarantees via Theorem 1. This avoids verifying the full MoE as a monolithic network, reducing verification cost and enabling incremental

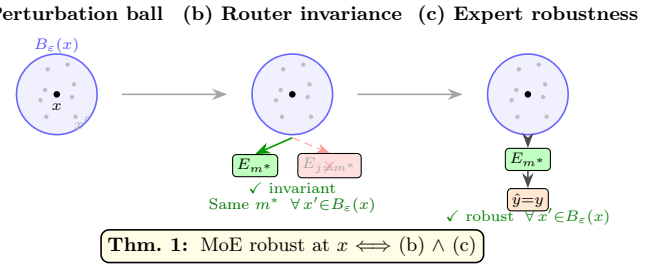

Fig. 2. Compositional robustness decomposition (Theorem 1). **(a)** Input x within perturbation ball $B_\varepsilon(x)$. **(b)** Router invariance: all perturbed inputs route to E_{m^*}. **(c)** Expert robustness: E_{m^*} predicts class y throughout the ball. The system is robust iff both hold.

certification – when a new expert is added, only the new expert and the updated router require (re-)verification, while existing expert certificates remain valid. Note that the biconditional holds under hard routing with disjoint class spaces; for overlapping classes only sufficiency holds, and soft/top-k routing requires verifying invariance of the full top-k set and stability of routing weights. For visually similar domains (*e.g.*, GTSRB [35] vs. PTSD [24], both traffic sign datasets), gating accuracy drops to 93.17%, so the router invariance premise is violated at ∼7% of inputs; we revisit this limitation in Section 6.6.

3.4 Empirical Results Supporting the Formal Findings

ExCRA and AA play complementary but distinct roles. ExCRA is the primary decision criterion: it provides a provable guarantee via formal verification. AA under PGD [22] serves only as supporting empirical evidence. At a fixed ε, soundness ensures ExCRA $\leq$ AA, but in practice the two metrics use different perturbation budgets ($\varepsilon = 2/255$ for ExCRA vs. $8/255$ for AA), so their absolute values are not directly comparable. We therefore examine whether empirical results preserve the same *model ordering* observed under certification, rather than treating AA as a substitute for ExCRA.

Both metrics aggregate per-image binary outcomes over a test set: "AA = 17%" means PGD found successful attacks on 83% of images, and "ExCRA = 90%" means the verifier formally certified 18 of 20 images. Prior work confirms that adversarial training generally improves certified robustness [32,2,21]; our experiments further reveal that non-robust training on complex domains yields complete verification failure (ExCRA = 0%).

4 Methodology

Why heterogeneous vision MoE? Deployed vision systems rarely stay within a single training distribution: handling inputs that span disjoint domains is

common and dispatching each input to a dataset-specific expert is therefore a natural design. We evaluate on both a visually-dissimilar pair (CIFAR-10 vs. MNIST) and a structurally-similar one (GTSRB+PTSD).

We independently train N expert networks $E_i : \mathbb{R}^{C\times H\times W} \to \mathbb{R}^{C_i}$, one per dataset and frozen after training, together with a single router $\mathcal{G} : \mathbb{R}^{C\times H\times W} \to \mathbb{R}^N$. We evaluate on four datasets: CIFAR-10 [19] (10 classes), MNIST [20] (10 classes), GTSRB [35] (German traffic signs, 43 classes), and PTSD [24] (Persian traffic signs, 43 classes). CIFAR-10 and MNIST form a visually dissimilar pair; GTSRB and PTSD share visual structure, testing the router under harder conditions. The framework extends to additional traffic sign datasets for lifelong learning (Appendix A). Each expert is denoted E_{i_NRT} or E_{i_RT} for non-robust and robust training respectively, with $i \in \{0{:}\mathrm{CIFAR\text{-}10}, 1{:}\mathrm{MNIST}, 2{:}\mathrm{GTSRB}, 3{:}\mathrm{PTSD}\}$.

4.1 Verification-Aware CNN Architecture

Both experts and router use a compact convolutional architecture (VerifiableCNN) designed for α–β-CROWN's bound propagation. Verification cost grows superlinearly with network size and nonlinearity count [38], and VNN-COMP benchmarks [16] confirm that current complete verifiers scale to CNNs of ∼100K parameters within practical timeouts. We therefore target this regime with 4 convolutional layers ($20 \to 28 \to 40 \to 56$ channels), 2×2 average pooling in blocks 1–3, and a 2-layer fully connected head (Appendix B).

Three design choices support LP-based verification: average pooling replaces max pooling to preserve linearity, increased channel capacity compensates for narrow widths, and static shapes minimize ONNX operators. Experts include BatchNorm (∼96K params for 43 classes), folded into convolutional weights during ONNX export (Appendix C). The router omits BatchNorm for deterministic verification behavior (∼38K params).

We train experts independently using Adam [18] (lr 10^{-3}, batch 128, 200 epochs, early stopping) with label-smoothing cross-entropy [36] (smoothing = 0.1) and standard augmentation under two paradigms: NRT with standard cross-entropy, and RT with PGD adversarial training [22,42] ($\varepsilon = 8/255$, 7 steps, step $\alpha = 2/255$).

4.2 Router (GatingNet)

The router $\mathcal{G}(x) \in \mathbb{R}^N$ produces N raw logits and selects expert $m^* = \arg\max_i g_i$. It shares the expert's convolutional backbone but omits BatchNorm, with its final layer outputting N routing logits instead of C_i class logits.

Because the router receives images from multiple datasets with different pixel distributions, all inputs are normalized using dataset-averaged statistics: $\mu_{\text{unified}} = \frac{1}{N}\sum_{i=0}^{N-1}\mu_i$, $\sigma_{\text{unified}} = \frac{1}{N}\sum_{i=0}^{N-1}\sigma_i$. This prevents distribution mismatch and ensures no single dataset dominates routing features.

Once all experts are frozen, the router is trained on meta-class labels $m \in \{0,\ldots,N{-}1\}$ using Adam (lr 10^{-3}, weight decay 5×10^{-4}, batch 128, 50 epochs,

early stopping). Under NRT, the objective is cross-entropy on clean inputs:

$$\mathcal{L}_{\text{NRT}} = \mathbb{E}_{(x,m)}\big[\text{CE}\big(\mathcal{G}(x),\, m\big)\big]. \tag{5}$$

RT adds an adversarial term, defining $x^{\text{adv}} = \arg\max_{x' \in B_\varepsilon(x)} \text{CE}(\mathcal{G}(x'), m)$:

$$\mathcal{L}_{\text{RT}} = \mathcal{L}_{\text{NRT}} + \mathbb{E}\big[\text{CE}\big(\mathcal{G}(x^{\text{adv}}),\, m\big)\big]. \tag{6}$$

4.3 Training Paradigms and MoE Composition

When composing the full MetaMoE system, training optimizes a combined objective $\mathcal{L} = \mathcal{L}_{\text{cls}} + \lambda \mathcal{L}_{\text{gate}}$, where $\mathcal{L}_{\text{cls}} = \text{CE}(\hat{y}, y)$ is the cross-entropy between the unified MoE output and the true class label, $\mathcal{L}_{\text{gate}} = \text{CE}(\mathcal{G}(x), m)$ is the cross-entropy between the router's output and the true meta-class, and $\lambda > 0$ is a scalar weight balancing the two terms. This dual loss structure allows the router to learn accurate dataset-level routing while the frozen experts maintain their classification performance.

5 Compositional Verification with α–β-CROWN

Standard α–β-CROWN verifies monolithic networks. Verifying our full MetaMoE system (router + 2 experts, ∼231K parameters) monolithically is *structurally infeasible* with current tools for two reasons. First, under PyTorch 2.1.0's trace-based ONNX exporter [29], export records a single execution path, hard-coding the router's expert selection into a static graph [28]. The ONNX `If` operator requires the same number of branch outputs and compatible output types [23,28], which complicates composition when experts have different class-space dimensions. Second, all current verifiers – bound propagation (CROWN [41], α–β-CROWN [38]), SMT-based (Marabou [40]), and abstract-interpretation (DeepPoly [34]) – require fixed computational graphs; no VNN-COMP benchmark (2021–2025) has included input-dependent routing [16]. Our compositional decomposition sidesteps both barriers by reducing the problem to standard feedforward CNN verification. This motivates a **compositional verification strategy**:

Key insight. Router correctness is independent of expert computations. We verify the router maintains invariant expert selection within ε-balls, then verify each expert independently. By Theorem 1, if both components are verified, the full system is verified:

$$\text{Verified}_{\text{Router}}(x \to i) \wedge \text{Verified}_{\text{Expert}_i}(x \to j) \implies \text{Verified}_{\text{MoE}}(x \to j) \tag{7}$$

where $\text{Verified}_{\text{Router}}(x \to i)$ means the router provably selects expert i for all $x' \in B_\varepsilon(x)$, and $\text{Verified}_{\text{Expert}_i}(x \to j)$ means expert i provably predicts class j for all $x' \in B_\varepsilon(x)$.

This reduces the verification target from the full system (∼231K parameters) to the largest single component (∼96K per expert or ∼38K for the router). Per-component cost is *constant* regardless of system size, yielding a ∼2.4× reduction at N=2 that scales linearly: ∼5× at N=5, ∼10× at N=10.

5.1 Adapting Verification to the MoE Setting

Applying α–β-CROWN to our compositional setting requires several adaptations. By Theorem 1, we export only the router as a standalone ONNX model, a $\sim 6\times$ reduction from the full system. Perturbation bounds are applied in the unified normalization space so that the verified ε matches the space used during adversarial training.

Two architectural decisions further support verification. First, the router omits BatchNorm, eliminating train/eval discrepancies while maintaining 99.97% routing accuracy. Second, because α–β-CROWN searches for counterexamples, we negate the desired property in the VNNLIB specification: for an input belonging to expert i, we assert that some other expert's logit exceeds i's. If no counterexample is found, the router is provably invariant.

6 Evaluation

We evaluate MetaMoE along three axes: formal verification with α–β-CROWN, empirical adversarial robustness with PGD attacks, and whether the empirical results support the formal findings. We then assess scalability.

6.1 Experimental Setup

All experiments use VerifiableCNN experts and router, trained under NRT or RT. Results are averaged over 5 independent runs on an NVIDIA GeForce RTX 4060 (8GB) with PyTorch 2.1.0 (CUDA 12.1), α–β-CROWN (commit d4c79e3), and fixed seed 42.

Test sets. Formal verification uses 20–40 correctly classified images per configuration ($\sim$11s/router sample, up to 300s/expert sample). PGD attacks evaluate the full test set of each dataset.

Experiments. We run four groups: (1) router verification at $\varepsilon \in \{2/255, 4/255, 8/255\}$ under NRT and RT (40 samples); (2) expert verification across 4 variants and 3 ε budgets (20 samples each); (3) empirical PGD robustness ($\varepsilon = 8/255$) for all 8 MoE configurations (40 total runs); and (4) AA vs. ExCRA ordering comparison.

6.2 Formal Verification

We used α–β-CROWN [38,41,16] to certify provable robustness (300s timeout; "unknown" indicates timeout, not violation).

Router verification. Figure 3 shows results for 40 samples (20 CIFAR-10, 20 MNIST) at three ε levels. Both NRT and RT routers achieve 100% RoCRA at $\varepsilon \leq 4/255$. At $\varepsilon = 8/255$, RT maintains 100% while NRT drops to 60% due to timeouts – zero falsifications occurred in any configuration. Both routers satisfy the invariance condition of Theorem 1 at standard perturbation budgets.

RoCRA ($\approx 100\%$) substantially exceeds end-to-end AA ($\approx 35\%$), which is expected: RoCRA verifies only routing invariance, while AA measures end-to-end

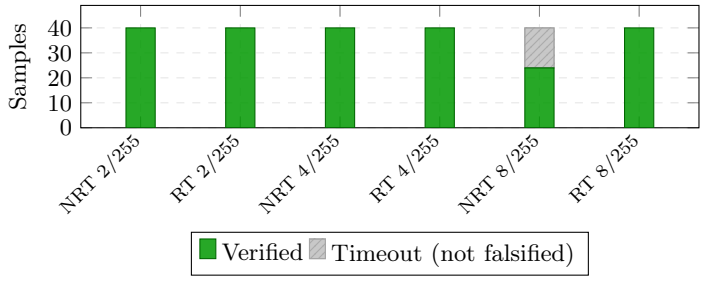

Fig. 3. Router formal verification (40 samples per configuration: 20 MNIST + 20 CIFAR-10, averaged over 5 runs). Both routers achieve 100% RoCRA at practical $\varepsilon \leq 4/255$. At $\varepsilon = 8/255$, only NRT degrades – and exclusively due to verification timeouts, not falsifications. Zero counterexamples were found in any configuration.

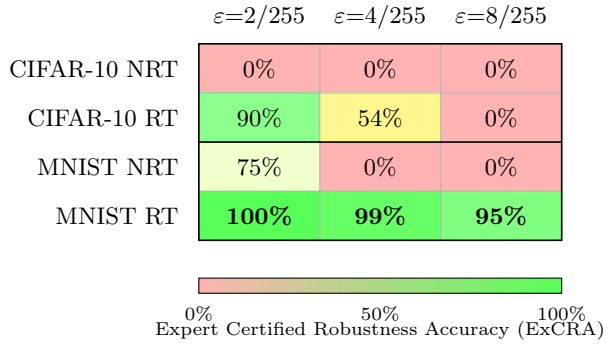

Fig. 4. ExCRA across domains, training paradigms, and perturbation budgets (20 samples per configuration, mean over 5 runs). Green indicates high ExCRA; red indicates verification failure (timeout). Robust training (RT) is a prerequisite for verifiability: CIFAR-10 NRT is completely unverifiable at all ε, while MNIST RT maintains $\geq$95% ExCRA even at $\varepsilon = 8/255$. Zero falsifications occurred in any configuration.

classification where expert misclassifications dominate. The router maintains 100% gating accuracy under PGD, but the selected expert may misclassify. The apparent inversion of ExCRA $\leq$ AA arises because RoCRA and AA use different ε values and measure different properties.

Expert verification. We verified CIFAR-10 and MNIST experts (20 samples each, 5 runs) using ONNX models with BatchNorm folding. Figure 4 reveals two patterns. First, domain complexity dominates verifiability: MNIST-RT achieves 100%/99%/95% ExCRA at $\varepsilon = 2/255, 4/255, 8/255$, while CIFAR-10-NRT is completely unverifiable (0% at all ε). Second, robust training is a prerequisite: CIFAR-10-RT achieves 90% ExCRA at $\varepsilon = 2/255$ vs. 0% for NRT, confirming that adversarial training tightens CROWN bounds. All failures are timeouts, not falsifications.

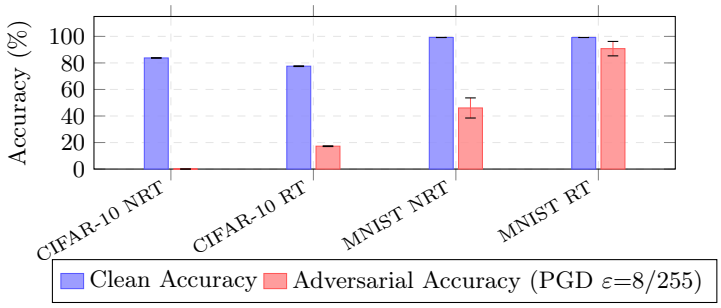

Fig. 5. Expert robustness–accuracy tradeoff (VerifiableCNN, mean ± std over 5 runs). The gap between blue (clean) and red (adversarial) bars represents the robustness gap. RT narrows the gap substantially for both domains, with MNIST achieving near-parity (8.39% gap) while CIFAR-10 retains a large gap (60.30%), reflecting domain complexity.

Table 1. Expert configuration dominates system robustness. *Clean Acc*: end-to-end MoE classification accuracy on unperturbed test inputs. *Adv Acc*: end-to-end MoE accuracy under PGD attack ($\varepsilon = 8/255$, 7 steps). *Robustness Gap*: Clean Acc − Adv Acc (smaller is better). Rows show expert training combinations averaged across both RT and NRT router training (which differ by $< 0.1\%$). 40 total runs: 8 configurations $\times$ 5 runs. All configurations achieve 100% gating accuracy.

Expert Configuration	**Clean Acc (%)**	**Adv Acc (%)**	**Robustness Gap**
Both RT	87.98 ± 0.06	41.93 ± 0.41	46.05%
CIFAR10-NRT + MNIST-RT	91.26 ± 0.05	37.66 ± 0.66	53.60%
CIFAR10-RT + MNIST-NRT	88.10 ± 0.06	33.56 ± 0.90	54.54%
Both NRT	91.32 ± 0.06	28.78 ± 0.59	62.54%
Expert effect: $\Delta_{\text{adv}} = 13.15\%$ (Both RT vs Both NRT)			
Router effect: $\Delta_{\text{adv}} < 0.1\%$ (RT Router vs NRT Router, 3.2× slower)			

6.3 Empirical Adversarial Robustness

We use PGD attacks (ε=8/255, 7 iterations, step 2/255) via the Adversarial Robustness Toolbox (ART).

Individual expert robustness. Figure 5 confirms the robustness–accuracy tradeoff: CIFAR-10 trades 6.21% clean accuracy for 17.23% AA gain; MNIST trades only 0.02% for 44.71% gain.

MetaMoE compositional robustness. We tested all 8 combinations of router training (RT/NRT) $\times$ 4 expert configurations, each trained 5 times (40 runs). Table 1 shows that expert configuration dominates: the 13.15% AA gap between Both-RT and Both-NRT far exceeds the $< 0.1\%$ router effect. All 40 runs achieve 100% gating accuracy under clean and adversarial conditions, validating Theorem 1's invariance condition. The weakest expert constrains system robustness, and NRT router training is 3.2× faster with identical accuracy.

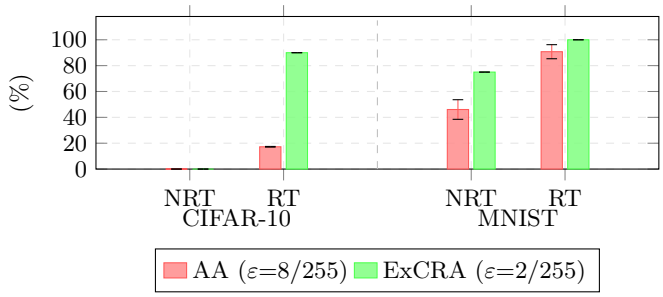

Fig. 6. Expert empirical results supporting the formal findings (AA at $\varepsilon = 8/255$, ExCRA at $\varepsilon = 2/255$; mean $\pm$ std over 5 runs). Because the perturbation budgets differ ($4\times$ larger ε for AA), absolute values are not directly comparable – in particular, ExCRA can exceed AA simply because it certifies a smaller perturbation ball. The key observation is that within each domain, the NRT→RT ordering is consistent across both metrics, indicating that the empirical robustness trends support the formal certification findings. CIFAR-10 NRT is completely unverifiable (ExCRA = 0%). Router results are omitted because RoCRA and router AA measure different properties.

GTSRB+PTSD achieves only 93.17% gating accuracy and 29.47% AA, compared to 100% and 41.93% for CIFAR10+MNIST (Both RT), demonstrating that routing difficulty propagates directly to system robustness. Standard deviations across runs remain below 0.08%, indicating highly reproducible outcomes.

6.4 Empirical Results Supporting the Formal Findings

Figure 6 compares expert AA (PGD, $\varepsilon = 8/255$) with ExCRA (α–β-CROWN, $\varepsilon = 2/255$). Because the perturbation budgets differ, we compare *orderings* rather than absolute values. Within each domain, the same ranking holds: MNIST NRT→RT increases both AA (46% → 91%) and ExCRA (75% → 100%), while CIFAR-10 NRT→RT moves from complete verification failure (ExCRA = 0%) to ExCRA = 90%. Routers are excluded because RoCRA and router AA measure different properties.

Empirical attacks do not replace certification, but they support the formal findings: within each domain, stronger empirical robustness corresponds to stronger certified robustness, while non-robustly trained models on complex domains fail to certify entirely.

6.5 Scalability

Training time. Figure 7 compares MetaMoE against monolithic training for 2-dataset training (GTSRB + CIFAR-10) and incremental addition of MNIST.

For 2 datasets (top pair), costs are comparable (91.5 vs. 99.5 mins) because both approaches must train on the same data. The advantage emerges when scaling: the bottom pair shows that adding MNIST forces the monolithic model

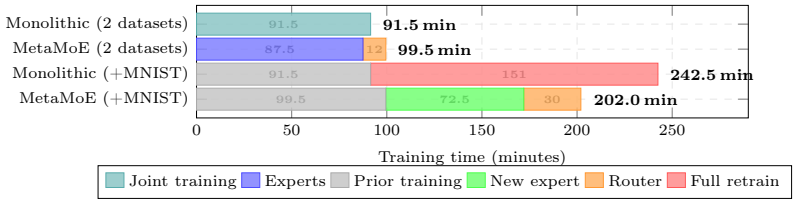

Fig. 7. Training time: monolithic vs. MetaMoE for initial 2-dataset training (top) and incremental MNIST addition (bottom). Top pair: costs are comparable (91.5 vs. 99.5 min). Bottom pair: gray segments show prior training carried over from the top pair; monolithic must then fully retrain on all three datasets (red, 151 min), while MetaMoE only trains the new MNIST expert (green, 72.5 min) and fine-tunes the router (orange, 30 min). The incremental cost drops from 151 min to 102.5 min, a 32% reduction.

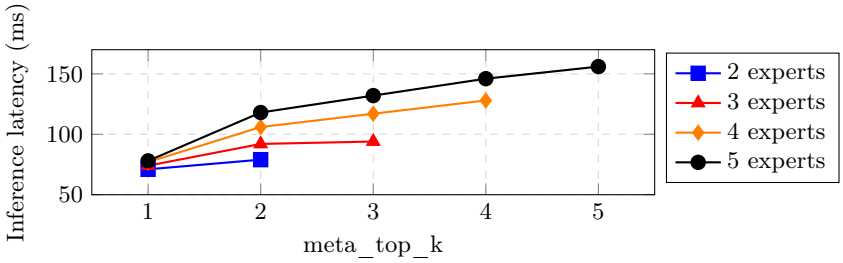

Fig. 8. Inference latency (ms) vs. meta_top_k for different numbers of total experts.

to retrain all three datasets from scratch (gray initial + red retrain = 242.5 mins), whereas MetaMoE carries over the full prior 2-dataset cost (blue, 99.5 mins), trains only the new MNIST expert independently (green, 72.5 mins), and fine-tunes the router (orange, 30 mins). The incremental cost is therefore 151 mins (monolithic) vs. 102.5 mins (MetaMoE), a 32% reduction. Crucially, only the 30-min router fine-tuning step touches the existing system; the 72.5-min expert training is fully independent and parallelizable. This gap widens with each additional expert, and the same linear scaling applies to re-verification ($\sim$2.4$\times$ at N=2, $\sim$7$\times$ at N=7).

Inference scaling. Latency follows $L(k) = L_{\text{router}} + \sum_{i=1}^{k} L_{\text{expert}_i}$. Figure 8 confirms $O(k)$ scaling: at k=1, latency is $\sim$71ms regardless of total expert count.

6.6 Limitations

Theorem 1 is established for hard routing (k=1) with disjoint class spaces. Soft or top-k routing requires invariance of the full top-k set and routing-weight stability; overlapping class spaces weaken the result to a one-sided implication (Section 3.3). The framework also depends on router discrimination: CIFAR-10 vs. MNIST gives 100% gating accuracy, while visually similar GTSRB vs. PTSD drops to 93.17%, violating invariance at $\sim$7% of inputs.

Our experiments use small verification-compatible CNNs on 20–40 samples per configuration (≈11s/router, up to 300s/expert); larger experts or sample budgets remain bounded by current tool capacity. We consider only ℓ_∞ perturbations; ℓ_2 and patch attacks need separate specifications.

Scope of the certified guarantee. Theorem 1 certifies each tested cube $B_\varepsilon(x)$ independently, not the input distribution, this is a *cube-wise* guarantee (Casadio et al.'s constraint security [3]) shared by all current complete verifiers [16] and is silent about unseen images. PGD-AA [22] provides the empirical complement.

7 Conclusion and Future Work

We presented MetaMoE, a framework for compositional robustness verification of heterogeneous MoE systems. Theorem 1 reduces system-level verification of hard-routed MoE with disjoint expert class spaces to the largest single component, with the advantage scaling linearly in the number of experts. Combining α–β-CROWN certification with empirical adversarial evaluation, we find that expert robustness dominates system behavior (13.15% AA gap between all-RT and all-NRT experts) while the router training paradigm is near-irrelevant ($< 0.1\%$), enabling a 3.2× router training speedup by forgoing adversarial augmentation. Router verification achieves 100% RoCRA at $\varepsilon \leq 4/255$ across all 40 runs, satisfying the invariance condition of Theorem 1.

We see four directions for extending this work. First, generalizing Theorem 1 to soft and top-k routing will require reasoning about weight stability under perturbation; probabilistic verification methods [4] may offer relaxed but practical guarantees in that setting. Second, extending these empirical results to more complex domains will require tighter bound propagation or certified training methods such as CROWN-IBP [43] integrated into the expert training pipeline. Third, the GTSRB+PTSD results (93.17% gating accuracy vs. 100% for CIFAR10+MNIST) expose a richer problem than routing failure alone: when experts share semantically equivalent classes, a misroute may not cause a misclassification. Relaxing Theorem 1 to account for cross-expert redundancy would yield a more permissive compositional guarantee for similar-domain systems.

Fourth, the same view may scale to MoE LLMs such as Mixtral [13], DeepSeek-MoE [6], and DeepSeek-V4 [8], which route each token to a sparse subset of hundreds of experts: a port of Theorem 1 would require the soft-routing extension above, distributional rather than argmax specifications [3,4], and per-expert verifiers that scale to transformer blocks [16].

The compositional perspective introduced here provides a foundation for building verifiable modular AI in safety-critical applications. The code is available: `https://github.com/PMQ9/Mixture-of-Experts_Research`.

Acknowledgments. The material presented in this paper is based upon work supported by the National Science Foundation (NSF) through grant number 2220401, the Defense Advanced Research Projects Agency (DARPA) under contract number FA8750-23-C-0518, and the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, under Award Number DE-SC0025528. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of DARPA or NSF.

A Dataset and Expert Notation

Table 2. Dataset and expert notation. Each expert is trained in two paradigms: Non-Robust Training (NRT) and Robust Training (RT).

Dataset	Meta-class	Notation	
		NRT	RT
CIFAR-10	E_0	E_{0_NRT}	E_{0_RT}
MNIST	E_1	E_{1_NRT}	E_{1_RT}
GTSRB	E_2	E_{2_NRT}	E_{2_RT}
PTSD	E_3	E_{3_NRT}	E_{3_RT}
BTSD	E_4	E_{4_NRT}	E_{4_RT}
ETSD	E_5	E_{5_NRT}	E_{5_RT}
TSRD	E_6	E_{6_NRT}	E_{6_RT}

B Detailed Architecture Specifications

The VerifiableCNN architecture used for experts (∼96K parameters with BatchNorm, 43 classes; ∼94K for 10 classes) and router (∼38K parameters without BatchNorm):

Convolutional Feature Extraction: Conv Block 1: Conv2d($3 \to 20$, 3×3, padding=1), ReLU, AvgPool(2×2) $\to 20 \times 16 \times 16$. Conv Block 2: Conv2d($20 \to 28$, 3×3, padding=1), ReLU, AvgPool(2×2) $\to 28 \times 8 \times 8$. Conv Block 3: Conv2d($28 \to 40$, 3×3, padding=1), ReLU, AvgPool(2×2) $\to 40 \times 4 \times 4$. Conv Block 4: Conv2d($40 \to 56$, 3×3, padding=1), ReLU $\to 56 \times 4 \times 4$.

Classification Head: Flatten($56 \times 4 \times 4 = 896$), Linear($896 \to 64$), ReLU, Dropout(0.3), Linear($64 \to C$) where $C = C_i$ for experts or $C = N$ for router. Output: raw logits (no softmax).

Expert vs. Router: Experts include BatchNorm after each convolution (∼96K params for 43 classes); the router omits BatchNorm for deterministic verification behavior (∼38K params for N=2).

Design Rationale: Gradual channel growth ($3 \to 20 \to 28 \to 40 \to 56$) compensates for narrow channels. Average pooling (linear operation) facilitates LP-based verification. Router omits BatchNorm to eliminate train/eval discrepancies. All 3×3 convolutions enable uniform bound propagation.

C ONNX Export Details

C.1 BatchNorm Folding

For a convolutional layer with weights $W \in \mathbb{R}^{C_{\text{out}} \times C_{\text{in}} \times k \times k}$ (where C_{out} is the number of output channels, C_{in} the number of input channels, and $k \times k$ the kernel size) and bias $b \in \mathbb{R}^{C_{\text{out}}}$, followed by a BatchNorm layer with running mean μ_i, running variance σ_i^2, learnable scale γ_i, and shift β_i (all per output channel i), with numerical stability constant $\varepsilon_{\text{bn}} = 10^{-5}$, the folded (merged) parameters are:

$$W'_{i,j,m,n} = \gamma_i \cdot \frac{W_{i,j,m,n}}{\sqrt{\sigma_i^2 + \varepsilon_{\text{bn}}}} \quad \forall i, j, m, n \tag{8}$$

$$b'_i = \frac{\gamma_i}{\sqrt{\sigma_i^2 + \varepsilon_{\text{bn}}}} \cdot (b_i - \mu_i) + \beta_i \quad \forall i \tag{9}$$

Here, the index i ranges over output channels ($1 \leq i \leq C_{\text{out}}$), j over input channels, and m, n over spatial kernel dimensions. The folded layer $\text{Conv}(W', b')$ is mathematically equivalent to $\text{BatchNorm}(\text{Conv}(W, b))$, eliminating the BatchNorm operator from the ONNX graph.

Verification: $\max |f_{\text{original}}(x) - f_{\text{folded}}(x)| < 10^{-5}$ for random inputs. All models exported with PyTorch 2.0+, ONNX opset 13, static input shape $(1, 3, 32, 32)$, raw logits output.

D Dataset Characteristics

Table 3. Dataset statistics.

Dataset	Classes	Train	Test	Domain
CIFAR-10	10	50,000	10,000	General objects
MNIST	10	60,000	10,000	Handwritten digits
GTSRB	43	39,209	12,630	German traffic signs
PTSD	43	14,405	2,421	Persian traffic signs
BTSD	62	4,575	2,520	Belgian traffic signs
ETSD	55	6,133	2,027	European traffic signs
TSRD	58	4,170	1,994	Chinese traffic signs

E Training Time Breakdown

Tables 4 and 5 show the full training time comparison between monolithic and MetaMoE compositional training.

Table 4. Training time comparison (2 datasets: GTSRB + CIFAR-10).

	Monolithic	**MetaMoE (2 experts)**
Training time	91.5 mins	99.5 mins
Clean accuracy	93.15%	92.80%
Details	Combined training	GTSRB: 33.5 mins
		CIFAR-10: 54 mins
		Router: 15 mins

Table 5. Training time comparison (3 datasets: + MNIST, incremental addition).

	Monolithic	**MetaMoE (3 experts)**
Training time	242.5 mins	202.0 mins
Clean accuracy	94.97%	93.61%
Details	Initial 2-dataset: 91.5 mins	Initial 2-expert MoE: 99.5
	Retrain to add MNIST: 151 mins	mins
		Add MNIST expert: 72.5 mins
		Fine-tune router: 30 mins

The key insight is the incremental cost: adding a new dataset requires 151 mins of full retraining for the monolithic model vs. 102.5 mins (72.5 + 30) for MetaMoE. Of the MetaMoE cost, only the 30-min router fine-tuning step touches the existing system; the 72.5-min expert training is fully independent and parallelizable.

References

1. Alsmann, E., Lange, M.: On the complexity of formal reasoning in state space models. In: International Symposium on AI Verification (SAIV) (2025)
2. Balunović, M., Vechev, M.: Adversarial training and provable defenses: Bridging the gap. In: International Conference on Learning Representations (2020)
3. Casadio, M., Komendantskaya, E., Daggitt, M.L., Kokke, W., Katz, G., Amir, G., Refaeli, I.: Neural network robustness as a verification property: A principled case study. In: International Conference on Computer Aided Verification (CAV). Lecture Notes in Computer Science, vol. 13371, pp. 219–231. Springer (2022)
4. Chwialkowski, M., Goubault, E., Putot, S.: Probabilistic verification of neural networks with sampling-based probability box propagation. In: International Symposium on AI Verification (SAIV) (2025)
5. Cohen, J., Rosenfeld, E., Kolter, Z.: Certified adversarial robustness via randomized smoothing. In: International Conference on Machine Learning (2019)

6. Dai, D., Deng, C., Zhao, C., Xu, R.X., Gao, H., Chen, D., Li, J., Zeng, W., Yu, X., Wu, Y., Xie, Z., Li, Y.K., Huang, P., Luo, F., Ruan, C., Sui, Z., Liang, W.: DeepSeekMoE: Towards ultimate expert specialization in mixture-of-experts language models. arXiv preprint arXiv:2401.06066 (2024)
7. DeepSeek-AI: DeepSeek-V3 technical report. arXiv preprint arXiv:2412.19437 (2024)
8. DeepSeek-AI: DeepSeek-V4: Towards highly efficient million-token context intelligence. arXiv preprint (2026), mixture-of-Experts LLM with 1.6T parameters (49B activated)
9. Fedus, W., Zoph, B., Shazeer, N.: Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. Journal of Machine Learning Research **23**(120), 1–39 (2022)
10. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. In: International Conference on Learning Representations (2015)
11. Henzinger, T.A., Qadeer, S., Rajamani, S.K.: You assume, we guarantee: Methodology and case studies. In: International Conference on Computer Aided Verification (1998)
12. Jacobs, R.A., Jordan, M.I., Nowlan, S.J., Hinton, G.E.: Adaptive mixtures of local experts. Neural Computation **3**(1), 79–87 (1991)
13. Jiang, A.Q., Sablayrolles, A., Roux, A., Mensch, A., Savary, B., Bamford, C., Chaplot, D.S., de las Casas, D., Hanna, E.B., Bressand, F., et al.: Mixtral of experts. arXiv preprint arXiv:2401.04088 (2024)
14. Jordan, M.I., Jacobs, R.A.: Hierarchical mixtures of experts and the EM algorithm. Neural Computation **6**(2), 181–214 (1994)
15. Katz, G., Barrett, C., Dill, D.L., Julian, K., Kochenderfer, M.J.: Reluplex: An efficient SMT solver for verifying deep neural networks. In: International Conference on Computer Aided Verification (2017)
16. Kaulen, K., Ladner, T., Bak, S., Brix, C., Duong, H., Flinkow, T., Johnson, T.T., Koller, L., Manino, E., Nguyen, T.H., Wu, H.: The 6th international verification of neural networks competition (VNN-COMP 2025): Summary and results. arXiv preprint arXiv:2512.19007 (2025)
17. Kielhoefer, L., Bosman, A.W., Hoos, H.H., van Rijn, J.N.: Robustness margin: A new measure for the robustness of neural networks. In: International Symposium on AI Verification (SAIV) (2025)
18. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: International Conference on Learning Representations (2015)
19. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images. Tech. rep., University of Toronto (2009)
20. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86**(11), 2278–2324 (1998)
21. Li, L., Xie, T., Li, B.: SoK: Certified robustness for deep neural networks. IEEE Symposium on Security and Privacy pp. 1289–1310 (2023)
22. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A.: Towards deep learning models resistant to adversarial attacks. In: International Conference on Learning Representations (2018)
23. ONNX Steering Committee: ONNX intermediate representation specification. `https://github.com/onnx/onnx/blob/main/docs/IR.md` (2025), graphs are topologically ordered, side-effect-free computations with frozen attributes.
24. Parsaseresht, S.: Persian traffic sign dataset (PTSD). `https://www.kaggle.com/datasets/saraparsaseresht/persian-traffic-sign-dataset-ptsd` (2021)

25. Pavlitska, S., Fan, H., Ditschuneit, K., Zöllner, J.M.: Robust experts: The effect of adversarial training on CNNs with sparse mixture-of-experts layers. arXiv preprint arXiv:2509.05086 (2025)
26. Pfeiffer, J., Ruder, S., Vulić, I., Ponti, E.M.: Modular deep learning. Transactions on Machine Learning Research (2023)
27. Puigcerver, J., Jenatton, R., Riquelme, C., Awasthi, P., Bhojanapalli, S.: On the adversarial robustness of mixture of experts. In: Advances in Neural Information Processing Systems (2022)
28. PyTorch Contributors: Export a model with control flow to ONNX — PyTorch tutorials. `https://pytorch.org/tutorials/beginner/onnx/export_control_flow_model_to_onnx_tutorial.html` (2025), conditional logic requires `torch.cond()` refactoring; both branches must produce the same number of outputs with compatible types.
29. PyTorch Contributors: torch.onnx — PyTorch documentation. `https://pytorch.org/docs/stable/onnx.html` (2025), "This exporter runs your model once in order to get a trace of its execution to be exported; if your model is dynamic, the export won't be accurate."
30. Ramasesh, V.V., Lewkowycz, A., Dyer, E.: Effect of scale on catastrophic forgetting in neural networks. In: International Conference on Learning Representations (2022)
31. Riquelme, C., Puigcerver, J., Mustafa, B., Neumann, M., Jenatton, R., Susano Pinto, A., Keysers, D., Houlsby, N.: Scaling vision with sparse mixture of experts. In: Advances in Neural Information Processing Systems. vol. 34, pp. 8583–8595 (2021)
32. Salman, H., Li, J., Razenshteyn, I., Zhang, P., Zhang, H., Bubeck, S., Yang, G.: Provably robust deep learning via adversarially trained smoothed classifiers. In: Advances in Neural Information Processing Systems. vol. 32 (2019)
33. Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., Dean, J.: Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In: International Conference on Learning Representations (2017)
34. Singh, G., Gehr, T., Püschel, M., Vechev, M.: An abstract domain for certifying neural networks. In: Proceedings of the ACM on Programming Languages (POPL). vol. 3, pp. 1–30 (2019)
35. Stallkamp, J., Schlipsing, M., Salmen, J., Igel, C.: Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. Neural Networks **32**, 323–332 (2012)
36. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 2818–2826 (2016)
37. Tran, H.D., Yang, X., Manzanas Lopez, D., Musau, P., Nguyen, L.V., Xiang, W., Bak, S., Johnson, T.T.: NNV: The neural network verification tool for deep neural networks and learning-enabled cyber-physical systems. In: International Conference on Computer Aided Verification (CAV). Lecture Notes in Computer Science, vol. 12224, pp. 3–17. Springer (2020)
38. Wang, S., Zhang, H., Xu, K., Lin, X., Jana, S., Hsieh, C.J., Kolter, J.Z.: Beta-CROWN: Efficient bound propagation with per-neuron split constraints for neural network robustness verification. In: Advances in Neural Information Processing Systems. vol. 34, pp. 29909–29921 (2021)
39. Watson, C., Alur, R., Gopinath, D., Mangal, R., Pasareanu, C.S.: Scenario-based compositional verification of autonomous systems with neural perception. In: International Symposium on AI Verification (SAIV) (2025)

40. Wu, H., Isac, O., Zeljić, A., Tagomori, T., Daggitt, M., Kokke, W., Refaeli, I., Amir, G., Julian, K., Bassan, S., Huang, P., Lahav, O., Wu, M., Zhang, M., Komendantskaya, E., Katz, G., Barrett, C.: Marabou 2.0: A versatile formal analyzer of neural networks. In: International Conference on Computer Aided Verification (CAV). Lecture Notes in Computer Science, vol. 14682, pp. 249–264. Springer (2024)
41. Xu, K., Shi, Z., Zhang, H., Wang, Y., Chang, K.W., Huang, M., Kailkhura, B., Lin, X., Hsieh, C.J.: Automatic perturbation analysis for scalable certified robustness and beyond. In: Advances in Neural Information Processing Systems. vol. 33 (2020)
42. Zhang, H., Yu, Y., Jiao, J., Xing, E., El Ghaoui, L., Jordan, M.: Theoretically principled trade-off between robustness and accuracy. In: International Conference on Machine Learning. pp. 7472–7482 (2019)
43. Zhang, H., Chen, H., Xiao, C., Gowal, S., Stanforth, R., Li, B., Boning, D., Hsieh, C.J.: Towards stable and efficient training of verifiably robust neural networks. In: International Conference on Learning Representations (2020)
44. Zhang, X., Xu, K., Hu, Z., Wang, R.: Optimizing robustness and accuracy in mixture of experts: A dual-model approach. In: International Conference on Machine Learning (2025)
45. Zhong, Y., Tan, S.Z.Z., Xu, H., Khoo, S.C.: GRENA: GPU-aided abstract refinement for neural network verification. In: International Symposium on AI Verification (SAIV) (2025)
46. Zhou, Y., Tripakis, S.: Compositional inductive invariant based verification of neural network controlled systems. In: NASA Formal Methods: 16th International Symposium, NFM 2024. Lecture Notes in Computer Science, vol. 14627, pp. 239–255. Springer (2024)