- **Random Sampling**: Points on the subproblem are randomly sampled to estimate upper bounds.

- **Fast Bounds:** method or technique for quickly estimating the bounds on the variables within a given optimization problem.

- **Longest edge** : divide the input domain by selecting the dimension or direction with the largest extent (i.e., the longest edge) and creating two subdomains based on this dimension.

- **imb:** Stands for intermediate bounds, which are estimates of the lower and upper bounds during the optimization process.

- **LP Solver:** a software tool/ algorithm, designed to solve the linear programming prob.

- **Approximation strategies in neural networks (NN)** are used to handle complex computations, particularly in situations where exact calculations are computationally expensive or infeasible.

- **"CollisionDetection"** dataset- In this dataset, the goal is to predict whether two vehicles following parameterized trajectories are likely to collide or not.

- This prediction is based on various features and properties.
- **Feature Extraction:** To make predictions, the dataset has 500 properties extracted from the problems. These properties are derived from a binary search process.
- **Network Architecture:** The neural network "**relatively shallow."** - it has fewer layers or parameters
- It's worth noting that even with a shallow network, the complexity of the dataset can make the decision boundaries challenging to determine.
- **Satisfiable (SAT) and Unsatisfiable (UNSAT):** In the context of this dataset, "satisfiable" refers to properties for which a counterexample can be found. In other words, if the prediction is false (indicating that a collision is predicted), it is

considered satisfiable because a counterexample can prove the prediction wrong. "Unsatisfiable" properties are those for which no counterexample exists, meaning the prediction is true (no collision is predicted).

- **Accuracy of Methods:** The text mentions that the results presented in Figure 4a highlight the accuracy of the methods used in predicting collisions. In this context, "methods" likely refers to the various techniques and algorithms employed for making these predictions, including the use of neural networks and the 500 extracted properties. The accuracy of these methods is a measure of how well they perform in correctly predicting whether collisions will occur or not.

- **Airborne Collision Avoidance System (ACAS) dataset**, used for a specific task related to aviation safety. The neural network in this dataset is deeper, which presents scalability challenges for the evaluation of various algorithms and methods.
- Airborne Collision Avoidance System (ACAS)**:** ACAS is a system designed to enhance the safety of aircraft by providing recommendations for horizontal maneuvers to avoid potential collisions. It does so based on sensor measurements and other relevant data. The dataset is likely derived from this system.
- **Neural Network-Based Advisory System:** The dataset uses a neural network as the basis for its advisory system. This neural network is responsible for providing recommendations regarding the choice of horizontal maneuvers to prevent collisions.
- **Maneuvers and Scores:** The neural network assesses the situation and assigns a score to each of the five possible maneuvers that an aircraft can undertake to avoid a collision. The assigned scores reflect the desirability or effectiveness of each maneuver.
- **The recommendation is then made based on the maneuver with the minimum score, which is considered the most suitable action to take.**
- **Properties to Verify:** The dataset includes 188 properties that need to be verified. These properties are likely related to specifications or conditions describing various scenarios in which the advisory system is expected to provide safe and effective recommendations.

- **Robust MNIST Network:** The network architecture consists of 2 convolution layers followed by 2 fully connected layers, resulting in a total of 4804 activation nodes.
- **Verification challenge** : The primary task related to the Robust MNIST Network is to verify the robustness of the network's predictions.
- In this context, robustness involves evaluating whether the predicted label for each input image changes when the image is perturbed within a specific ε-infinity norm ball. This disturbances assesses how resilient the network's predictions are to slight variations in the input images.
- In order to know the computation there is a time limit on 1 hr for both the MNIST networks.

- **Reduced Robust MNIST Network:** A simplified version of the Robust MNIST Network is presented in order to assess the effectiveness of a particular branching heuristic.
- this reduced network has the same structure as the original one but with fewer hidden nodes on each hidden layer, resulting in a total of 1226 ReLU nodes.
- **Verification :** Different ε values are used for both networks to assess the difficulty level of the verification properties. A larger ε value implies a more significant disturbance of the input image. The choice of ε values is adjusted based on the network's robustness, with a higher ε used for the larger network, which is expected to be more robust.

PCAMNIST and TwinStream

- More extensive evaluation of verification methods.
- Existing datasets used for verification do not provide the flexibility to explore the impact of various problem and model parameters like such as the depth of neural networks, the number of hidden units, input dimensionality, and the correlation between hidden nodes on the same layer
- designed to address the limitations of existing datasets for testing and evaluating neural network verification methods.

- **PCAMNIST:** This dataset primarily serves the purpose of evaluating methods over different network architectures. It offers a wide range of variation in terms of network depth (number of layers), the number of hidden units in each layer, and input dimensionality. However, it does not introduce specific correlations between nodes on the same layer.
- **TwinStream:** hidden nodes on the same layer are highly correlated.
- This dataset is tailored to explore the trade-off between different bounding strategies in the context of correlated hidden nodes.
- It offers a unique opportunity to study the impact of such correlations on verification methods.

ANALYSIS

Analysis - To evaluate the performance of different methods on various datasets

SMALL NETWORKS

- 4a of CollisionDetection, most solvers succeed against all properties in about 10 seconds.
- Planet, Reluplex and the MIP - are extremely fast.
- reluBaB - much slower than blackbox
- All methods solve quickly in the time gap.

- in Figure 4b, no errors are observed but most methods timeout on the most challenging testcases.
- The best method - Reluplex, which reaches 79.26% success rate at the two hours
- BaBSB, already achieves 98.40% with a budget of one hour.
- To reach Reluplex's success rate, the required runtime is two orders of magnitude smaller.

- Contains a limited number of verification properties or scenarios that need to be assessed for accuracy and reliability.

- objective : evaluate the trade-off between two key factors:
- 1. the cost associated with bounding
- 2. total number of branching decisions required during the verification process.
- 2 factors that have influence are :
- **Figure 5a:** displays a graphical representation of the trade-off. It shows the number of subdomains (regions or areas within the network's input space) that must be explored before the verification of properties is completed.
- Fewer subdomains indicate a more efficient verification process.
- **Table 3:** The text mentions that Table 3 provides information on the average time cost associated with exploring each subdomain. This suggests that not only is the number of subdomains important, but the time required to investigate each subdomain also matters.
- These times indicate the efficiency of each method in terms of how quickly they can analyze or explore individual nodes as part of their respective algorithms.
- BaBSB significantly reduces the number of nodes that need to be visited when compared to a method that does not utilize smart branching which is BaB.

LARGE NETWORKS

- Provides set of methods and tools for analyzing the behavior and vulnerabilities of neural networks.
- The performance of various methods is evaluated on the **reduced Robust MNIST Network** and the **Robust MNIST Network**, both of which **involve a large number of properties to cover a range of difficulties.**
- MIPplanet is considered the benchmark, and methods that couldn't perform at least as well as MIPplanet on simple properties are excluded.
- ERAN stands for "ETH Robustness Analyser for neural networks.” - tool for finding the robustness of the NN
- ERAN - interpretation-based tool for neural network verification
- Planet, ReluBaB, Reluplex, and Neurify couldn't solve any properties within a 100-second time limit.
- BaBSB managed to solve some properties but was not as effective as MIPplanet.

- BaBSR and ERAN performed as well as MIPplanet and were tested on all properties of large networks.
- In these comprehensive tests, **BaBSR performed the best** among all methods on both data sets.

CONCLUSION

- The study presents a unified Branch-and-Bound framework for analyzing and improving existing methods for neural network verification
- The newly proposed methods have shown significant performance improvements on comprehensive datasets.
- **Bounding:** Intermediate bounds are values that help assess the behavior of neural networks during verification. Tighter intermediate bounds provide a more accurate understanding of the network's behavior but typically require more computational resources to compute.
- **Branching**: Branching strategies are crucial for deciding how to split verification problems into smaller sub-problems, and the current methods are mostly heuristic-based. However, these heuristics might not be flexible enough to adapt when the nature of the verification problem changes.
- **"Offline costs":** refer to the computational overhead and time required for the initial training of these heuristics.
- Cheaper yet focused strategies should be remained focus.
- LPs are frequently solved to decide whether a branching should occur on a subproblem.
- Learning to imitate LP decision-making could significantly accelerate the entire process.