We wish to thank the reviewers for the comments. We will fix all the issues related to the clarity of presentation.

Reviewer 1

Q: In dataset NCI1, the gap between upper and lower bound certificate is larger for linear activation than ReLU activation.

A: In general, it is true that the gap for linear activation should be smaller than that for ReLU activation. This has been the case for all datasets, except when $s = 2$ and 3 for NCI1 (Figure 11 and 12). We double checked the experiment and did not find an error. Since the convex envelop is still a lower bound of the true objective $F_c(A)$, there could be exceptions in some cases for some datasets.

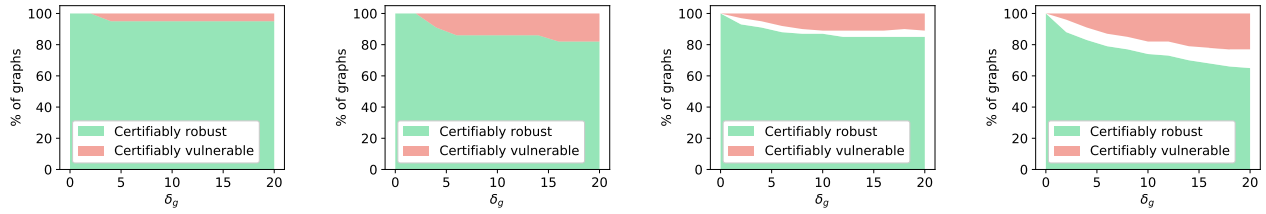
Reviewer 2

Q: A more intuitive explanation of polar operator should be provided.

A: We will add more details on the conditional gradient algorithm (a.k.a. Frank–Wolfe, `https://en.wikipedia.org/wiki/Frank%E2%80%93Wolfe_algorithm`) and the polar operator (step 1 of the algorithm on the wikipedia page). The projected gradient descent algorithm is not feasible in our context because projection to co(A) is hard and even explicitly expressing co(A) in terms of linear constraints can be hard. To bypass this difficulty, the conditional gradient algorithm was adopted, which instead resorts to maximizing a linear function over co(A) (polar operator). This is equivalent to maximizing a linear function over A, and can be solved efficiently as shown in Section 4.1.

Q: using 30% of the data for training

A: We additionally experimented on Enzyme with 80%, 10%, 10% for training, validation and testing, respectively. The small size of test data led to marked variations in the gap plot, and it is unclear how to "average" them. So we plotted a typical result below, which shows the resulting fraction of certifiably robust / vulnerable for both linear and ReLU activations. Compared with Figure 7 and 8 in the Supplementary material where 30% graphs were used for training, the tightness here appears similar, or slightly better under the linear activation.



(a) Linear ($s = 2$) (b) Linear ($s = 3$) (c) ReLU ($s = 2$) (d) ReLU ($s = 3$)

Q: What are the training algorithms? What is the robust training?

A: The non-robust (NR) training objective is empirical risk minimization on Eq (1), where ℓ is the cross entropy loss. The robust training (R) follows exactly from [11, 12], where a hinge loss (line 264) is added to Eq (1) that encourages a larger prediction margin. A detailed description is provided in the second paragraph of Section 5 (line 263 to 269). Both objectives were optimized by Adam.

Reviewer 3

Thank you for your comment. We will improve the organization of related works.

Reviewer 4

Q: Can the method scale to graphs with a larger number of nodes and edges?

A: We examined another dataset DD [1], where the median #node $= 284$ and median #edge $= 716$ per graph. We set $\delta_g = 20$ and $s = 3$. For the certificates with linear activation, Enzyme (median #node $= 32$, median #edge $= 120$) takes 0.37 seconds per graph on average, while DD takes 7.3 seconds. For the certificates with ReLU activation, Enzyme takes 1 second per graph on average, while DD takes 28 seconds. This is consistent with the fact that the computational cost depends quadratically on the number of nodes (in practice, a bit lower than that due to implementation details).

Different datasets have different number of classes. To facilitate comparison, the reported time cost is for each class (see $\min_c$ in line 102.5). The number of edges in the original graph does not affect the computational cost much, because the attacker can both delete **and** add edges.

[1] Benchmark Data Sets: `https://ls11-www.cs.tu-dortmund.de/staff/morris/graphkerneldatasets`