Convex Bounds on the Softmax Function with Applications to Robustness Verification

Dennis Wei
IBM Research

Haoze Wu
Stanford University

Min Wu
Stanford University

Pin-Yu Chen
IBM Research

Clark Barrett
Stanford University

Eitan Farchi
IBM Research

Abstract

The softmax function is a ubiquitous component at the output of neural networks and increasingly in intermediate layers as well. This paper provides convex lower bounds and concave upper bounds on the softmax function, which are compatible with convex optimization formulations for characterizing neural networks and other ML models. We derive bounds using both a natural exponential-reciprocal decomposition of the softmax as well as an alternative decomposition in terms of the log-sum-exp function. The new bounds are provably and/or numerically tighter than linear bounds obtained in previous work on robustness verification of transformers. As illustrations of the utility of the bounds, we apply them to verification of transformers as well as of the robustness of predictive uncertainty estimates of deep ensembles.

1 INTRODUCTION

The softmax function is an indispensable component of multiclass classifiers ranging from multinomial logistic regression models to deep neural networks (NNs). It is most often deployed at the output of a classifier to convert K real-valued scores corresponding to K classes into a probability distribution over the classes. More recently, the softmax is playing an increasing role in intermediate layers as well with the popularization of Transformers (Vaswani et al., 2017), whose quintessential component, the (self-)attention mechanism (Luong et al., 2015; Gehring et al., 2017), utilizes softmax to compute attention scores.

Our main contribution in this paper is to provide convex bounds on the softmax function. More precisely, we derive lower bounds on the outputs of the softmax that are convex functions of the inputs, and upper bounds on the outputs that are concave functions of the inputs (see (4) later). This enables the formulation of convex optimization problems for characterizing ML models with softmax components, particularly in intermediate layers.

We apply our bounds to verification of the robustness of NNs against adversarial input perturbations. We consider in particular the quantification of predictive uncertainty for multiclass classifiers, which is typically assessed in terms of accurate estimation of the conditional probability distribution. We are not aware of prior work that directly addresses the robustness of uncertainty estimation metrics, especially for deep ensembles (Lakshminarayanan et al., 2017; Rahaman and Thiery, 2021), although the works of Bitterwolf et al. (2020); Berrada et al. (2021) are related.

Our results are summarized as follows. In Section 3, we first consider an exponential-reciprocal decomposition of the softmax function, used by Shi et al. (2020); Bonaert et al. (2021) in robustness verification of transformers. While Shi et al. (2020); Bonaert et al. (2021) limited themselves to linear bounds, we instead derive nonlinear convex bounds (which we refer to as "ER") and show that these are tighter than the previous linear bounds ("lin"). We then consider in Section 4 an alternative decomposition in terms of the log-sum-exp (LSE) function, a well-known convex function (Boyd et al., 2004), and obtain corresponding bounds. We prove that the LSE upper bound is always tighter than the ER upper bound, and that the LSE lower bound is tighter than the ER lower bound for the case of $K = 2$ inputs. These analytical results are summarized by the following inequalities, where $L(x)$ and $U(x)$ denote

Proceedings of the 26th International Conference on Artificial Intelligence and Statistics (AISTATS) 2023, Valencia, Spain. PMLR: Volume 206.

lower and upper bounds that are functions of the input x:

$$L^{\mathrm{lin}}(x) \leq L^{\mathrm{ER}}(x) \underbrace{\leq}_{K=2} L^{\mathrm{LSE}}(x) \leq \mathrm{softmax}(x) \leq U^{\mathrm{LSE}}(x) \leq U^{\mathrm{ER}}(x) \leq U^{\mathrm{lin}}(x). \quad (1)$$

For $K > 2$ inputs, while there are instances where $L^{\mathrm{ER}}(x) > L^{\mathrm{LSE}}(x)$, our numerical experiment in Section 6 suggests that this does not occur often and that in some regimes, L^{LSE} is tighter by factors of 3 or more in terms of the mean gap with respect to $\mathrm{softmax}(x)$. For the upper bounds, we find that $U^{\mathrm{ER}}(x)$ improves considerably upon $U^{\mathrm{lin}}(x)$, which can be rather loose, and $U^{\mathrm{LSE}}(x)$ consistently improves upon $U^{\mathrm{ER}}(x)$ by a further factor of 2. In Section 7, we describe experiments on robustness verification of transformers and of uncertainty estimation by deep ensembles. The results provide further evidence of the hierarchy in (1) and of the potential usefulness of the bounds.

1.1 Related Work

Deterministic robustness certification has gained increasing interest in the past few years (Katz et al., 2017; Gehr et al., 2018). Most of the work has focused on verifying properties of the pre-softmax output (e.g., Katz et al. (2017); Gehr et al. (2018)) for piecewise-linear NNs, while formal reasoning about the softmax outputs themselves has been under-explored. Katz et al. (2017) showed that even the pre-softmax verification problem is computationally inefficient (NP-complete), but a number of approaches based on (linear) abstraction (Singh et al., 2019c; Weng et al., 2018b; Zhang et al., 2018; Gowal et al., 2019) and convex optimization (Wong and Kolter, 2018; Dvijotham et al., 2018) have been proposed to strike a good balance between scalability and precision. We review existing work on NN verification more thoroughly in App. A.

As mentioned, the softmax function appears in intermediate layers of transformers, and previous works on robustness verification of transformers (Shi et al., 2020; Bonaert et al., 2021) have developed linear lower and upper bounds to approximate the softmax. We review the bounds of Shi et al. (2020); Bonaert et al. (2021) in Section 3.1 as a prelude to deriving provably tighter bounds.

Some works have addressed robustness verification of specifications that involve softmax outputs (probabilities) and not just softmax inputs. Bitterwolf et al. (2020) obtained an upper bound on the maximal probability (i.e., confidence) to verify the robustness of out-of-distribution detectors and train detectors with such guarantees. Their bound coincides with one of our constant bounds in Section 2.1. Berrada et al. (2021) proposed a general framework for probabilistic specifications on the softmax output, where the NN can be stochastic and inputs can have uncertainty. While the uncertainty quantification metrics that we consider fall under their framework, Berrada et al. (2021) do not give explicit formulations for them, let alone implemented algorithms.

Bounds on the softmax and log-sum-exp functions have been used in other contexts. For example, Titsias (2016) derived lower bounds on softmax motivated by large-scale classification, Bouchard (2007) investigated three upper bounds on log-sum-exp for approximate Bayesian inference, and Nielsen and Sun (2016) used bounds on log-sum-exp to bound information-theoretic measures of mixture models. These bounds, however, do not have the convexity/concavity that we require in the different cases.

2 PRELIMINARIES

For an input $x \in \mathcal{R}^K$, the output p of the softmax function is given by

$$p_j = \frac{e^{x_j}}{\sum_{j'=1}^{K} e^{x_{j'}}} = \frac{1}{1 + \sum_{j' \neq j} e^{x_{j'} - x_j}}, \quad j = 1, \ldots, K. \quad (2)$$

We work with the second form above, which is preferred in general for numerical stability and also by Bonaert et al. (2021) for facilitating their approximations (see their Sec. 5.2). To ease notation, we will focus on the first output p_1, without loss of generality because of symmetry. Table 6 summarizes the notation used in the paper. Based on the second form in (2), we accordingly define $\tilde{x}_j := x_j - x_1$, $j = 1, \ldots, K$. In the simplest case of $K = 2$, the softmax reduces to the logistic sigmoid:

$$p_1 = \frac{1}{1 + e^{\tilde{x}_2}} = 1 - p_2. \quad (3)$$

We assume that the set of inputs is contained in the hyperrectangle defined by $l_j \leq x_j \leq u_j$, $j = 1, \ldots, K$, which we write as $l \leq x \leq u$. Our goal is to obtain lower bounds $L(x)$ on p_1 that are convex functions of x, and upper bounds $U(x)$ that are concave functions of x,

$$L(x) \leq p_1 \leq U(x). \quad (4)$$

Constraints of the form in (4) are desirable in general because they define convex sets of (x, p_1) and can be incorporated into convex optimization problems.

2.1 Basic Bounds and Constraints

It can be seen from (2) that p_1 is strictly decreasing in $\tilde{x}_j = x_j - x_1$ for $j \neq 1$. We assume that we have bounds $\tilde{l} \leq \tilde{x} \leq \tilde{u}$ on these differences and also define $\tilde{l}_1 = \tilde{u}_1 := 0$ for the trivial case $\tilde{x}_1 = 0$. Given $l \leq x \leq u$, $\tilde{l}_j = l_j - u_1$ and $\tilde{u}_j = u_j - l_1$ are always valid bounds on $\tilde{x}_j$ for $j \neq 1$, but we may have tighter bounds as well. The constraints

$\tilde{l} \le \tilde{x} \le \tilde{u}$ lead to lower and upper bounds on p_1:

$$\underline{p}_1 = \frac{1}{1 + \sum_{j=2}^{K} e^{\tilde{u}_j}} = \frac{1}{\mathrm{SE}(\tilde{u})}, \tag{5a}$$

$$\overline{p}_1 = \frac{1}{1 + \sum_{j=2}^{K} e^{\tilde{l}_j}} = \frac{1}{\mathrm{SE}(\tilde{l})}, \tag{5b}$$

where we have defined the sum-of-exponentials function $\mathrm{SE}(x) := \sum_{j=1}^{K} e^{x_j}$. These are constant bounds in the sense that they are not functions of x. The upper bound (5b) coincides with the bound of Bitterwolf et al. (2020, eq. (6)) given the input bounds $\tilde{l}$.

We also have the property that p is non-negative and sums to 1:

$$\sum_{j=1}^{K} p_j = 1, \qquad p_j \ge 0 \quad \forall j, \tag{6}$$

which are linear constraints on p. Bonaert et al. (2021) recognized the benefit of explicitly enforcing (6) while Shi et al. (2020) did not use such a constraint.

3 BOUNDS FROM EXPONENTIAL -RECIPROCAL DECOMPOSITION

Previous work (Bonaert et al., 2021) took the natural approach of decomposing the softmax function (2) as the composition of a sum of exponentials, $q_1 = \mathrm{SE}(\tilde{x})$, and the reciprocal function $p_1 = 1/q_1$.[1] They as well as Shi et al. (2020) derive lower and upper bounds on $\mathrm{SE}(\tilde{x})$ and $1/q_1$ that are affine in x and q_1 respectively, and compose these bounds to obtain bounds on the softmax. We review the bounds of Shi et al. (2020); Bonaert et al. (2021) in Section 3.1, combining their respective advantages, before starting to improve upon them in Section 3.2.

3.1 Existing Linear Bounds

Sum of Exponentials For the sum of exponentials $\mathrm{SE}(\tilde{x})$, each exponential $e^{\tilde{x}_j}$ is a function of a scalar $\tilde{x}_j \in [\tilde{l}_j, \tilde{u}_j]$. By virtue of convexity and following Shi et al. (2020), each exponential can be bounded from above by the chord between the endpoints $(\tilde{l}_j, e^{\tilde{l}_j})$ and $(\tilde{u}_j, e^{\tilde{u}_j})$, and from below by a tangent line passing through (t_j, e^{t_j}). The resulting bounds can be written as

$$\mathrm{SE}(\tilde{x}) \ge 1 + \sum_{j=2}^{K} e^{t_j}(\tilde{x}_j - t_j + 1), \tag{7a}$$

$$\mathrm{SE}(\tilde{x}) \le \overline{\mathrm{SE}}(\tilde{x}; \tilde{l}, \tilde{u}), \tag{7b}$$

where

$$t_j = \min\left\{ \log \frac{e^{\tilde{u}_j} - e^{\tilde{l}_j}}{\tilde{u}_j - \tilde{l}_j}, \tilde{l}_j + 1 \right\} \tag{8}$$

and we have defined the chordal upper bound on $\mathrm{SE}(x)$,

$$\overline{\mathrm{SE}}(x; l, u) = \sum_{j=1}^{K} \left(\frac{u_j - x_j}{u_j - l_j} e^{l_j} + \frac{x_j - l_j}{u_j - l_j} e^{u_j} \right). \tag{9}$$

In (8), the first choice of t_j makes the slope e^{t_j} in the lower bound (7a) equal to the corresponding slope in the upper bound (7b), thus minimizing the area between them (Bonaert et al., 2021). The second term in (8) ensures that the lower bound (7a) is non-negative for all $\tilde{x}_j \in [\tilde{l}_j, \tilde{u}_j]$.[2] In (9), we adopt the convention that if $l_j = x_j = u_j = 0$ (as is true for $\tilde{l}_1, \tilde{x}_1, \tilde{u}_1$), then the jth term in the sum is 1.

Reciprocal The same approach is applied to the reciprocal $1/q_1$, which is also a convex function of a scalar. First we need lower and upper bounds on the input q_1 to the reciprocal. These are obtained by minimizing the lower bound (7a) and maximizing the upper bound (7b) over $\tilde{x} \in [\tilde{l}, \tilde{u}]$, resulting in

$$\underline{q}_1^{\mathrm{lin}} = 1 + \sum_{j=2}^{K} e^{t_j}(\tilde{l}_j - t_j + 1), \tag{10a}$$

$$\overline{q}_1^{\mathrm{lin}} = \mathrm{SE}(\tilde{u}) = \frac{1}{\underline{p}_1}. \tag{10b}$$

Then we have the following bounds on the reciprocal:

$$\frac{1}{t_{q_1}} \left(2 - \frac{q_1}{t_{q_1}} \right) \le \frac{1}{q_1} \le \frac{1}{\underline{q}_1^{\mathrm{lin}}} + \underline{p}_1 - \frac{\underline{p}_1 q_1}{\underline{q}_1^{\mathrm{lin}}}, \tag{11}$$

where $t_{q_1} = \max\{\sqrt{\underline{q}_1^{\mathrm{lin}} \overline{q}_1^{\mathrm{lin}}}, \overline{q}_1^{\mathrm{lin}}/2\}$ is the q_1 value of the tangent point.

Softmax Overall bounds on the softmax output p_1 are obtained by composing bounds (7) and (11) with $q_1 = \mathrm{SE}(\tilde{x})$ and $p_1 = 1/q_1$. Specifically, upper bound (7b) is composed with the lower bound in (11) to yield the overall lower bound

$$L^{\mathrm{lin}}(\tilde{x}) = \frac{1}{t_{q_1}} \left(2 - \frac{\overline{\mathrm{SE}}(\tilde{x}; \tilde{l}, \tilde{u})}{t_{q_1}} \right). \tag{12a}$$

Similarly, the combination of (7a) and the upper bound in (11) yield

$$U^{\mathrm{lin}}(\tilde{x}) = \frac{1}{\underline{q}_1^{\mathrm{lin}}} + \underline{p}_1 - \frac{\underline{p}_1}{\underline{q}_1^{\mathrm{lin}}} \left(1 + \sum_{j=2}^{K} e^{t_j}(\tilde{x}_j - t_j + 1) \right). \tag{12b}$$

[1] Bonaert et al. (2021)'s use of the second form in (2) avoids a multiplication needed by Shi et al. (2020).

[2] In this second case, we allow the slopes in (7a), (7b) to be different, like Shi et al. (2020) and unlike Bonaert et al. (2021).

3.2 New Nonlinear Bounds

We now depart from Shi et al. (2020); Bonaert et al. (2021) and derive nonlinear bounds on the softmax function using the same exponential-reciprocal decomposition. This is done by further exploiting the convexity of the functions $\mathrm{SE}(\tilde{x})$ and $1/q_1$.

Sum of Exponentials We now regard q_1 as an intermediate variable corresponding to the sum of exponentials $\mathrm{SE}(\tilde{x})$ but no longer bound by the strict equality $q_1 = \mathrm{SE}(\tilde{x})$. For a lower bound on q_1, we use $\mathrm{SE}(\tilde{x})$ itself, i.e., we relax $q_1 = \mathrm{SE}(\tilde{x})$ to

$$q_1 \geq \mathrm{SE}(\tilde{x}). \tag{13}$$

Since $\mathrm{SE}(\tilde{x})$ is convex, the above constraint is in the desired form as in (4): it specifies a convex set of (x, q_1) and is compatible with convex optimization. On the other hand, for an upper bound on q_1, we require a concave function of x. We thus reuse the upper bound (7b), which is linear (and hence concave) in x.

Reciprocal Similarly for the reciprocal where the exact relation is $p_1 = 1/q_1$, we use $1/q_1$ itself as the lower bound on p_1 and reuse the upper bound in (11). For the latter however, it is possible to substitute a tighter lower bound on q_1 than $\underline{q}_1^{\mathrm{lin}}$ in (10a). The reason is that we can now minimize the lower bound in (13) over $\tilde{x} \in [\tilde{l}, \tilde{u}]$ instead of the one in (7a), resulting in

$$\underline{q}_1^{\mathrm{ER}} = \mathrm{SE}(\tilde{l}) = \frac{1}{\overline{p}_1}. \tag{14}$$

The upper bound on q_1 is still $\overline{q}_1^{\mathrm{lin}}$ (10b). We therefore have

$$\frac{1}{q_1} \leq p_1 \leq \overline{p}_1 + \underline{p}_1 - \overline{p}_1 \underline{p}_1 q_1. \tag{15}$$

Softmax Overall bounds on the softmax are obtained by composing (13), (7b) with (15), specifically the lower bound with the upper bound and vice versa. The results are

$$L^{\mathrm{ER}}(\tilde{x}) = \frac{1}{\overline{\mathrm{SE}}(\tilde{x}; \tilde{l}, \tilde{u})}, \tag{16a}$$

$$U^{\mathrm{ER}}(\tilde{x}) = \overline{p}_1 + \underline{p}_1 - \overline{p}_1 \underline{p}_1 \,\mathrm{SE}(\tilde{x}). \tag{16b}$$

The lower bound $L^{\mathrm{ER}}(\tilde{x})$ is a composition of $\overline{\mathrm{SE}}(\tilde{x}; \tilde{l}, \tilde{u})$, an affine function of x, with the reciprocal function. It is thus convex by the composition properties of convex functions (Boyd et al., 2004, Sec. 3.2.2). The upper bound $U^{\mathrm{ER}}(\tilde{x})$ has a sum of exponentials with a negative multiplier in front and is hence concave, as desired.

Theorem 1. *The nonlinear bounds $L^{\mathrm{ER}}(\tilde{x})$, $U^{\mathrm{ER}}(\tilde{x})$ are tighter than the linear bounds $L^{\mathrm{lin}}(\tilde{x})$, $U^{\mathrm{lin}}(\tilde{x})$:*

$$L^{\mathrm{lin}}(\tilde{x}) \leq L^{\mathrm{ER}}(\tilde{x}) \leq p_1 \leq U^{\mathrm{ER}}(\tilde{x}) \leq U^{\mathrm{lin}}(\tilde{x}) \quad \forall \tilde{x} \in [\tilde{l}, \tilde{u}].$$

We defer all proofs to Appendix C.

4 BOUNDS FROM LOG-SUM-EXP DECOMPOSITION

In this section, we depart from the exponential-reciprocal decomposition altogether and consider an alternative decomposition, obtained by taking the logarithm of the softmax (2), $r_1 = -\mathrm{LSE}(\tilde{x})$, and then exponentiating, $p_1 = e^{r_1}$. Here $\mathrm{LSE}(x) = \log\left(\sum_{j=1}^{K} e^{x_j}\right)$ is the "log-sum-exp" (LSE) function, a well-known convex function. Its negative $-\mathrm{LSE}(\tilde{x})$ is therefore concave in x.

We follow the same approach as in Section 3.2, bounding the exponential and LSE functions and then composing the bounds.

Exponential For the exponential function $p_1 = e^{r_1}$, which is again a convex function of a scalar, we use e^{r_1} itself as the lower bound on p_1 and a chord of the function as the upper bound. Noting that $\mathrm{LSE}(\tilde{x})$ is increasing in all inputs $\tilde{x}_j$, r_1 is bounded within the interval $[-\mathrm{LSE}(\tilde{u}), -\mathrm{LSE}(\tilde{l})]$, and we may thus use the chord connecting the points $(-\mathrm{LSE}(\tilde{u}), e^{\ -\mathrm{LSE}(\tilde{u})})$ and $(-\mathrm{LSE}(\tilde{l}), e^{\ -\mathrm{LSE}(\tilde{l})})$. Using (5) to rewrite $-\mathrm{LSE}(\tilde{u})$, $-\mathrm{LSE}(\tilde{l})$ as $\log(\underline{p}_1)$, $\log(\overline{p}_1)$, the bounds on p_1 in terms of r_1 are

$$e^{r_1} \leq p_1 \leq \frac{\log(\overline{p}_1) - r_1}{\log(\overline{p}_1) - \log(\underline{p}_1)} \underline{p}_1 + \frac{r_1 - \log(\underline{p}_1)}{\log(\overline{p}_1) - \log(\underline{p}_1)} \overline{p}_1. \tag{17}$$

Log-Sum-Exp For the log-sum-exp function $-\mathrm{LSE}(\tilde{x})$, since it is concave in x, we may use it as the *upper* bound on its output r_1:

$$r_1 \leq -\mathrm{LSE}(\tilde{x}). \tag{18}$$

It remains to find a lower bound on $-\mathrm{LSE}(\tilde{x})$ that is convex in x. In the case $K = 2$, $-\mathrm{LSE}(\tilde{x}) = -\log(1 + e^{\tilde{x}_2})$ is a concave function of a scalar $\tilde{x}_2 \in [\tilde{l}_2, \tilde{u}_2]$ and we may bound it as before using the chord between endpoints,

$$-\log\left(1 + e^{\tilde{x}_2}\right) \geq -\frac{\tilde{u}_2 - \tilde{x}_2}{\tilde{u}_2 - \tilde{l}_2} \log\left(1 + e^{\tilde{l}_2}\right) - \frac{\tilde{x}_2 - \tilde{l}_2}{\tilde{u}_2 - \tilde{l}_2} \log\left(1 + e^{\tilde{u}_2}\right). \tag{19}$$

This is linear and hence convex in x.

For $K > 2$, the challenge is that $-\mathrm{LSE}(\tilde{x})$ is a multivariate function. Here we provide two bounds with different strengths and weaknesses. In Appendix D, we describe a third bound that more directly extends the $K = 2$ case (19) but turns out not to be as tight. For the first bound, we rewrite $-\mathrm{LSE}(\tilde{x})$ as $-\mathrm{LSE}(\tilde{x}) = x_1 - \mathrm{LSE}(x)$ so that $-\mathrm{LSE}(x)$ is the non-convex part to be bounded. For this,

we use chordal bounds on the exponentials similar to (7b):

$$\mathrm{LSE}(x) \le \log\left(\overline{\mathrm{SE}}(x; l, u)\right). \tag{20}$$

Hence

$$-\mathrm{LSE}(\tilde{x}) \ge x_1 - \log\left(\overline{\mathrm{SE}}(x; l, u)\right). \tag{21}$$

For the second bound, let $j = \arg\max_j (l_j + u_j)$ be the index of the largest input in terms of the midpoints $m_j = (l_j + u_j)/2$. Define the vector of differences $\dot{x}$, $\dot{x}_j = x_j - x_{j^*}$, $j = 1, \ldots, K$, with corresponding lower and upper bounds $\dot{l}_j$, $\dot{u}_j$. We then use the relation $\tilde{x} = \dot{x} - \dot{x}_1$ and the translation property of LSE to write $-\mathrm{LSE}(\tilde{x}) = \dot{x}_1 - \mathrm{LSE}(\dot{x})$. By bounding $\mathrm{LSE}(\dot{x})$ using chordal bounds on exponentials as in (20), we obtain

$$-\mathrm{LSE}(\tilde{x}) \ge \dot{x}_1 - \log\left(\overline{\mathrm{SE}}(\dot{x}; \dot{l}, \dot{u})\right). \tag{22}$$

In both (21) and (22), since the term after x_1 or $\dot{x}_1$ is the composition of an affine function of x with $-\log$, the right-hand sides of (21), (22) are convex functions of x as desired. The advantage of (22) is that the inputs $\dot{x}_j$ to $\mathrm{LSE}(\dot{x})$ tend to be negative, thus placing them in the flatter parts of the exponentials $e^{\dot{x}_j}$ and leading to less loss when these exponentials are bounded by chords. The disadvantage of (22) is that the chords are over intervals $[\dot{l}_j, \dot{u}_j]$ that tend to be wider than the intervals $[l_j, u_j]$ used in (21).

Softmax Overall bounds on the softmax are obtained by composing (18) and (21) or (22) with (17), this time matching lower bound with lower bound and upper with upper:

$$L^{\mathrm{LSE}}(x) = \frac{e^{x_1}}{\overline{\mathrm{SE}}(x; l, u)}, \tag{23a}$$

$$L^{\mathrm{LSE}}(x) = \frac{e^{\dot{x}_1}}{\overline{\mathrm{SE}}(\dot{x}; \dot{l}, \dot{u})}, \tag{23b}$$

$$U^{\mathrm{LSE}}(x) = \frac{\underline{p}_1 \log(\overline{p}_1) - \overline{p}_1 \log(\underline{p}_1) - (\overline{p}_1 - \underline{p}_1)\,\mathrm{LSE}(\tilde{x})}{\log(\overline{p}_1) - \log(\underline{p}_1)}. \tag{23c}$$

We use the lower bounds in (23a), (23b) for $K > 2$ and the upper bound (23c) for all K. Bound (23b) is in fact a generalization of the ER lower bound (16a) and coincides with (16a) when $j = 1$. The lower bounds (23a), (23b) are the compositions of the right sides of (21), (22), previously argued to be convex in x, with the exponential function, which is convex and increasing. Hence $L^{\mathrm{LSE}}(x)$, $L^{\mathrm{LSE}}(x)$ are convex by the composition properties of convex functions (Boyd et al., 2004, Sec. 3.2.4). The upper bound $U^{\mathrm{LSE}}(x)$ is concave in x as it has $\mathrm{LSE}(\tilde{x})$ with a negative multiplier.

For an overall lower bound in the case $K = 2$, we take lower bound (19) instead of (21) or (22) and exponentiate. After simplifying, this yields

$$L^{\mathrm{LSE}_2}(x) = \left(\underline{p}_1\right)^{(\tilde{x}_2 \;\; \tilde{l}_2)/(\tilde{u}_2 \;\; \tilde{l}_2)} \left(\overline{p}_1\right)^{(\tilde{u}_2 \;\; \tilde{x}_2)/(\tilde{u}_2 \;\; \tilde{l}_2)}, \tag{24}$$

which is an exponential function of $\tilde{x}_2$ and hence convex (LSE_2 indicates that this bound is only for $K = 2$).

Theorem 2. *The log-sum-exp upper bound $U^{\mathrm{LSE}}(x)$ is tighter than the nonlinear exponential-reciprocal upper bound $U^{\mathrm{ER}}(x)$,*

$$p_1 \le U^{\mathrm{LSE}}(x) \le U^{\mathrm{ER}}(x) \quad \forall x \in [l, u],$$

for all $K \ge 2$. The log-sum-exp lower bound $L^{\mathrm{LSE}_2}(x)$ is tighter than the nonlinear exponential-reciprocal lower bound $L^{\mathrm{ER}}(x)$,

$$L^{\mathrm{ER}}(x) \le L^{\mathrm{LSE}_2}(x) \le p_1 \quad \forall x \in [l, u],$$

for $K = 2$.

For $K = 2$ inputs, the softmax function (3) and all bounds (linear, ER, LSE) can be plotted as functions of the scalar $\tilde{x}_2$. We do so in Figure 1 for the input interval $[\tilde{l}_2, \tilde{u}_2] = [-2, 2]$. In addition to confirming Theorems 1 and 2, the figure shows that the gap between the ER lower bound L^{ER} and softmax is about twice as large as for the LSE lower bound L^{LSE_2}, and similarly for the upper bounds. While L^{lin} is tangent to L^{ER}, U^{lin} exhibits a larger gap. These observations continue to hold for $K > 2$ in Section 6.

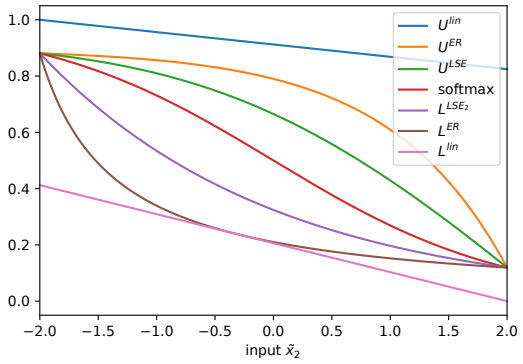


Figure 1: Linear (lin), exponential-reciprocal (ER), and log-sum-exp (LSE) lower and upper bounds on logistic sigmoid function (softmax for $K = 2$ inputs).

5 LINEARIZED BOUNDS

Any tangent plane to a convex lower bound is also a sound lower bound; and any tangent plane to a concave upper bound is also a sound upper bound. A plane tangent to a function $f : \mathcal{R}^K \mapsto \mathcal{R}$ at point c can be described by:

$$\overline{f}_c(x) = \sum_{j=1}^{K} \left(\frac{\partial f(c)}{\partial x_j} (x_j - c_j) \right) + f(c). \tag{25}$$

The coefficient of each addend $(x_j - c_j)$ is the partial derivative of $f(x)$ with respect to x_j evaluated at c. As an example, $\frac{\partial L^{\mathrm{ER}}(x)}{\partial x_j}$ can be computed with the chain rule and the results are:

$$\frac{\partial L^{\mathrm{ER}}(x)}{\partial x_1} = L^{\mathrm{ER}}(x)^2 \cdot \sum_{j=2}^{K} \left(\frac{e^{\tilde{u}_j} - e^{\tilde{l}_j}}{\tilde{u}_j - \tilde{l}_j} \right) \tag{26a}$$

$$\frac{\partial L^{\mathrm{ER}}(x)}{\partial x_j} = -L^{\mathrm{ER}}(x)^2 \cdot \frac{e^{\tilde{u}_j} - e^{\tilde{l}_j}}{\tilde{u}_j - \tilde{l}_j} \quad \text{for } j \neq 1. \tag{26b}$$

Note that while the above linearized bounds derived from L^{ER} and the existing linear bound described in Eq. (12a) are both based on the exponential-reciprocal decomposition, there is a key difference: (12a) is obtained by composing the linear over-approximations for each decomposition step, while our approach only linearizes *once* after composing the non-linear bounds.

Linearized bounds for the other nonlinear bounds in Sections 3.2 and 4 can be found in App. E.

While the tangent planes as bounds are strictly less tight than their nonlinear counterparts, these linearized bounds can be useful because they can be integrated in existing bound-propagation frameworks (Singh et al., 2019b; Zhang et al., 2018) for NNs, which are designed to efficiently bound the outputs of NNs given a set of inputs. We use them to verify self-attention mechanisms in Section 7.2

6 SYNTHETIC DATA EVALUATION

We conduct an experiment using synthetic data to compare the tightness of the bounds in Sections 3 and 4. For this experiment, we first sample softmax outputs from a K-dimensional Dirichlet distribution. To simulate outputs that have varying amounts of probability concentrated on one component, we choose one component of the mean, $\mu_{j_{\max}}$, to be larger than the others, $\mu_{j_{\max}} = \mu_{\max}$, and vary $\mu_{\max}$. More details are in App. F. We continue to focus on the first softmax output p_1 and consider two cases: $j_{\max} = 1$ (p_1 has the largest mean) and $j_{\max} \neq 1$ (p_1 is among those with small mean). After sampling a softmax output p, we convert it to an input m (i.e., logits). Bounds on the input region are then set as $l_j = m_j - \epsilon$, $u_j = m_j + \epsilon$ for all j, where the width ϵ is varied. Inputs x are sampled from the uniform distribution over the hypercube $[l, u]$. One hundred (100) input regions are generated in this manner, and from each region, 1000 inputs x are sampled.

For each input x, we evaluate p_1 (2) and the following lower and upper bounds: constant $\underline{p}_1$, $\overline{p}_1$ (5), linear (12), ER (16), and LSE (23), (24). We leave the linearized bounds of Section 5 to App. F. We compute the *mean gap* $p_1 - L(x)$ between the softmax and each lower bound, where the mean is taken over the uniform samples x, and similarly the mean gap $U(x) - p_1$ for each upper bound.

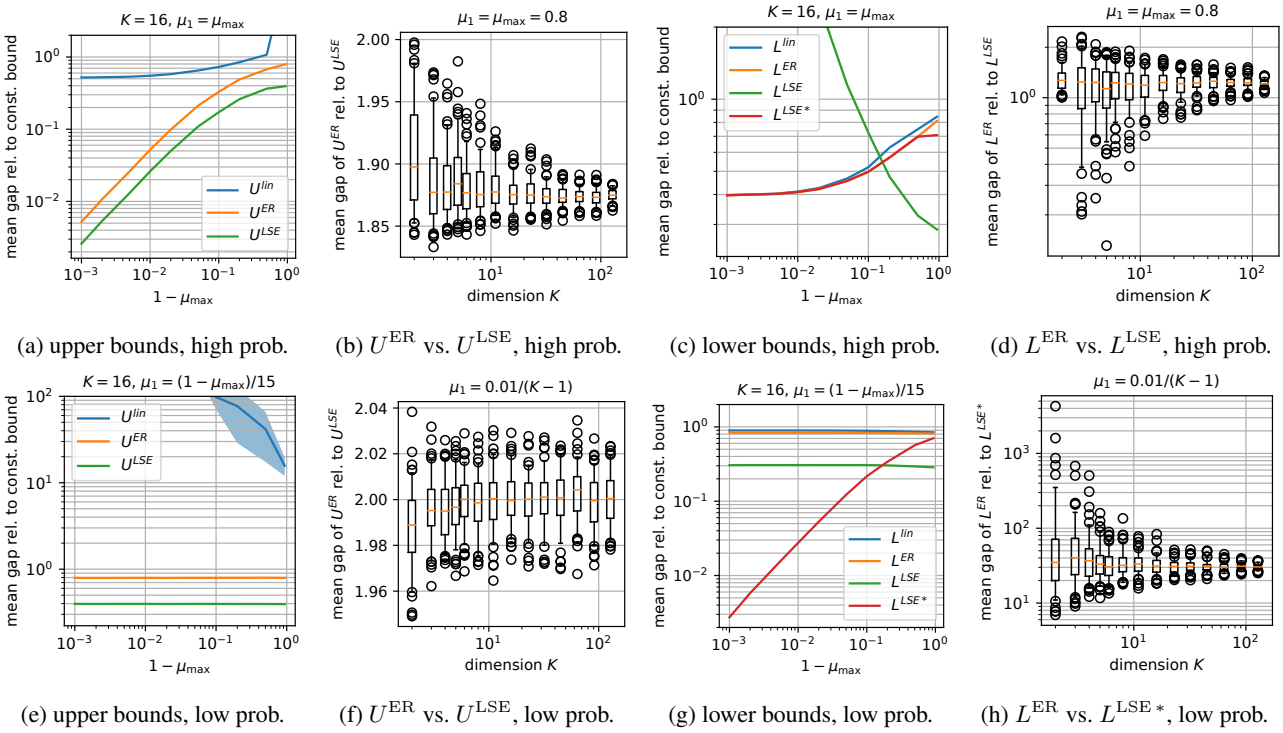
In Figure 2, we plot ratios of mean gaps to focus more on the comparisons between the various bounds. Plots of the mean gaps themselves are in App. F. In Figures 2a, 2e, for each input region we divide the mean gap of each upper bound by the mean gap of the constant bound (i.e., $\overline{p}_1 - p_1$). We then plot as a function of $\mu_{\max}$ the mean ratios, taken over the 100 input regions, as well as the standard errors in the mean. Figures 2c, 2g are the same for the lower bounds. In Figures 2b, 2f, we take the ratio of the mean gap of U^{ER} to that of U^{LSE} and show box plots over the 100 input regions (whiskers at the 5th and 95th percentiles) for different values of K. Figures 2d, 2h are the same for L^{ER} versus L^{LSE} and L^{ER} versus L^{LSE} respectively. The top row of Figure 2 represents the case $j_{\max} = 1$, where p_1 tends to be high, while the bottom row corresponds to $j_{\max} \neq 1$ and low p_1. App. F contains plots for values of K and ϵ other than those indicated in Figure 2.

We first discuss the upper bounds (left two columns of Figure 2). The linear ER bound U^{lin} can be quite loose, as previously suggested by Figure 1 (see App. F for a possible explanation). In Figure 2e, U^{lin} is worse than the constant bound $\overline{p}_1$ by at least an order of magnitude. Thus, moving to the novel nonlinear bound U^{ER} can already make a big difference. The bound U^{LSE} provides further improvement, as guaranteed by Theorem 2, and the improvement factor of around 2 is remarkably consistent as a function of $\mu_{\max}$ and K. This is particularly evidenced by the narrow distributions of ratios in Figures 2b, 2f.

Turning now to the lower bounds, L^{lin} is stronger than its counterpart U^{lin} in the sense that it improves upon the constant bound $\underline{p}_1$. The improvement from L^{lin} to the novel nonlinear bound L^{ER} is more marginal. The two LSE bounds L^{LSE} (23a) and L^{LSE} (23b) are indeed seen to be complementary as discussed in Section 4. For $\mu_{\max} \lesssim 0.8$, the largest softmax output does not tend to be that much larger than the others and L^{LSE} is better, whereas for $\mu_{\max} \gtrsim 0.9$, the largest component dominates and L^{LSE} is better. In the case $j_{\max} \neq 1$ in Figure 2g, the combination of L^{LSE} and L^{LSE} offer an improvement over L^{ER} by a factor ranging from 2.5–3 to much higher. However for $j_{\max} = 1$ and $\mu_{\max} \gtrsim 0.9$ in Figure 2c, L^{LSE} coincides with L^{ER} and there is no improvement. In Figures 2d, 2h, for $K = 2$ (leftmost box plot), we use (24) as the LSE lower bound and the box plots confirm the inequality $L^{\mathrm{ER}}(x) \leq L^{\mathrm{LSE}_2}(x)$ from Theorem 2. For $K > 2$, the median ratio of mean gaps (orange lines) remains approximately constant, although a minority of instances have a ratio less than 1 in Figure 2d.

7 APPLICATIONS TO ROBUSTNESS VERIFICATION

We present experiments on two robustness verification problems. Our focus remains on showing that the new



(a) upper bounds, high prob. (b) U^{ER} vs. U^{LSE}, high prob. (c) lower bounds, high prob. (d) L^{ER} vs. L^{LSE}, high prob.

(e) upper bounds, low prob. (f) U^{ER} vs. U^{LSE}, low prob. (g) lower bounds, low prob. (h) L^{ER} vs. $L^{\mathrm{LSE}*}$, low prob.

Figure 2: Mean gaps of upper bounds (left two columns) and lower bounds (right two columns) on softmax output p_1 for synthetically generated input regions of width $\epsilon = 1$. In the top/bottom row, the mean μ_1 of p_1 is high/low.

bounds in Sections 3 and 4 provide benefits for these tasks, in addition to their theoretical and numerical advantages.

7.1 Predictive Uncertainty Estimation

Accurately quantifying uncertainty in predictions is important for calibrating users and for identifying highly uncertain and out-of-distribution examples. Many solutions have been proposed for predictive uncertainty estimation with NNs. Here we focus on the popular technique of using a *deep ensemble* of NNs (Lakshminarayanan et al., 2017; Rahaman and Thiery, 2021). Verification of the robustness of deep ensembles has not been studied to our knowledge.

To measure the quality of uncertainty estimates, we consider two *proper scoring rules* (Gneiting and Raftery, 2007), negative log-likelihood (NLL) and Brier score. Given an instance $(x\ , y\ )$ with true label y (x now refers to the overall NN input) and predicted probabilities p_k for each class $k = 1, \ldots, K$, the scoring rule assigns a score $S(p, y\ )$. To verify the robustness of uncertainty estimates p, we bound the worst score that can be attained within an ℓ_p ball $\mathcal{B}_p(x\ , \epsilon)$ of radius ϵ around $x\ $. Using the convention that lower scores are better, we thus wish to solve

$$\max_{x 2 \mathcal{B}_p(x^*, \epsilon)} S(p, y\ ). \tag{27}$$

We formulate (27) as a concave maximization problem for tractability. Part of this involves expressing $S(p, y\ )$ as, or bounding it from above by, a concave function of p. This also suffices for a deep ensemble, where p is the average of probabilities $p^{(m)}$ from the models in the ensemble, since $S(p, y\ )$ or its upper bound will also be concave in $p^{(m)}$.

Negative Log-Likelihood In the case of NLL, the scoring rule is $S(p, y\ ) = -\log p_{y^*}$. While this is not concave in p, we can equivalently maximize the linear function

$$S(p, y\ ) = -p_{y^*}. \tag{28}$$

Brier Score Here the scoring rule is

$$S(p, y\ ) = \sum_{k=1}^{K} (p_k - \delta_{k=y^*})^2 = (1 - p_{y^*})^2 + \sum_{k \neq y^*} p_k^2, \tag{29}$$

which is a convex sum-of-squares function of p. For tractable optimization, we instead maximize an affine upper bound on the Brier score. For each softmax output p_k, we have constant bounds $\underline{p}_k \leq p_k \leq \overline{p}_k$ from (5) (generalized to all k, and averaged over models for a deep ensemble). We can then bound the convex univariate functions $(1-p_{y^*})^2$ and p_k^2 by the chords connecting their endpoints, as done throughout Sections 3 and 4. The resulting bound can be written as the following affine function of p:

$$S(p, y\ ) \leq -2p_{y^*} + \sum_{k=1}^{K} (\underline{p}_k + \overline{p}_k) p_k - \sum_{k=1}^{K} \underline{p}_k \overline{p}_k + 1. \tag{30}$$

Given one of the concave scoring objectives in (28) or (30), we relate p to the logits predicted by the network(s) using the lower and upper bounds in Sections 3 and 4. The logits play the role of x in these bounds. We then encode the remainder of the NN(s), from input to the logits, using existing convex relaxations, and specifically the triangular linear relaxation (Ehlers, 2017) in our experiment. For completeness, the full formulation of (27) as a concave maximization problem is provided in Appendix G.

For our experiment, we train two deep ensembles with 5 NNs each on the MNIST dataset using the Uncertainty Baselines[3] package in Python. The architecture of the deep ensemble is given in App. H. We consider ℓ_1-ball ($\mathcal{B}_1(x, \epsilon)$) perturbations in (27). Input values to the NNs are normalized to between $[0, 1]$. On this scale, we use perturbation bounds $\epsilon = 0.008, 0.012, 0.016$, which correspond to roughly 2, 3, and 4 pixel values. For each of the first 100 test images, (27) is solved to bound the score, either NLL or Brier, using different bounds on the softmax. We use CVXPY (Diamond and Boyd, 2016) and its included off-the-shelf solver SCS (O'Donoghue et al., 2021) to solve these concave problems. We then average over the test images to bound the expected score.

Table 1 shows the resulting bounds on the expected score. The "clean" values are those without perturbation, i.e., $\epsilon = 0$. Recall that the nonlinear bounds denoted ER and LSE (the two rightmost columns in Table 1) are both our contributions. While we wished to pair L^{ER} with U^{ER}, using the latter caused the SCS solver to not converge, so we substituted instead the stronger bound U^{LSE}. For the LSE pair, we used L^{LSE} since it suits typical softmax outputs. The results indicate that our new nonlinear bounds result in more precise verification of uncertainty quantification than existing linear bounds (represented by lin). In Appendix I.1, we complement these results with *lower* bounds on worst-case uncertainty estimation scores obtained by using a PGD attack, while in Appendix I.2, we show results for an ensemble of larger networks.

Table 1: Upper Bounds on Uncertainty Estimation Scores Using Different Softmax Bounds for the MNIST Classifier

Score (Clean)	ϵ	$L^{\mathrm{lin}}, U^{\mathrm{lin}}$	$L^{\mathrm{ER}}, U^{\mathrm{LSE}}$	$L^{\mathrm{LSE}*}, U^{\mathrm{LSE}}$
NLL (0.105)	2/256	0.265	0.261	**0.251**
	3/256	0.442	0.433	**0.420**
	4/256	0.726	0.697	**0.690**
Brier (0.048)	2/256	0.138	0.134	**0.131**
	3/256	0.244	0.235	**0.234**
	4/256	0.417	**0.403**	**0.403**

We then repeat the same experiment on an ensemble model trained on the CIFAR-10 dataset (architectures can be found in App. H). Results are shown in Table 2. The results again support the greater strength of the nonlinear bounds over the linear bounds, across ϵ values and scoring rules. Interestingly, in this case, the ER lower bound is superior to the LSE$*$ lower bound, suggesting that in practice, ER and LSE bounds can be complementary.

Table 2: Upper Bounds on Uncertainty Estimation Scores Using Different Softmax Bounds for the CIFAR-10 Classifier

Score (Clean)	ϵ	$L^{\mathrm{lin}}, U^{\mathrm{lin}}$	$L^{\mathrm{ER}}, U^{\mathrm{LSE}}$	$L^{\mathrm{LSE}*}, U^{\mathrm{LSE}}$
NLL (1.538)	2/256	2.118	**2.014**	2.028
	3/256	2.569	**2.433**	2.474
	4/256	3.087	**2.940**	3.013
Brier (0.690)	2/256	0.971	**0.917**	0.920
	3/256	1.170	**1.114**	1.120
	4/256	1.367	**1.324**	1.329

Timing Results The average analysis times in seconds per instance using different bounds are shown in Table 3. The experiments are performed on a cluster equipped with Intel Xeon E5-2637 v4 CPUs. Each job is given one CPU.

Table 3: Average Runtime in Seconds using Different Bounds

Dataset	$L^{\mathrm{lin}}, U^{\mathrm{lin}}$	$L^{\mathrm{ER}}, U^{\mathrm{LSE}}$	$L^{\mathrm{LSE}*}, U^{\mathrm{LSE}}$
MNIST	10.9	91.6	92.4
CIFAR-10	19.5	95.3	95.5

It must be noted that for this experiment, we are using CVXPY and its off-the-shelf solver SCS, which take time to convert the problems into standard forms and are not customized for them. Hence the runtime results should be interpreted only as confirmation that the convex problems are indeed tractable to solve. In Appendix I.3, we show results from a further over-approximation in which each network in the ensemble is considered separately, which improves computational efficiency.

7.2 Self-Attention Mechanisms

In addition, we consider verifying canonical adversarial robustness properties on neural networks with self-attention mechanisms (Vaswani et al., 2017). Self-attention layers involve not only the softmax function, but also bilinear transformations, which are non-trivial to encode as convex optimization problems. Therefore, in this task we instead leverage existing bound-propagation-based methods (Singh et al., 2019b; Zhang et al., 2018; Shi et al., 2020), which already handle bilinear constraints. In particular, we use the CROWN/DeepPoly framework (Singh et al., 2019b; Zhang et al., 2018), a popular bound-propagation method, which requires that each neuron is over-approximated with one linear upper bound and one

[3] Available from GitHub repository `https://github.com/google/uncertainty-baselines`

linear lower bound. Tight convex bounds for bilinear transformation are left as future work.

MNIST We train three NNs (named `A_small`, `A_med`, and `A_big`) with self-attention mechanisms on the MNIST dataset and consider verifying their adversarial robustness against l_1-norm bounded perturbations. The networks are all PGD-trained and vary in size due to different hyperparameters in the self-attention layers. Details can be found in App. H. We evaluate on the first 500 test images of the dataset. Inputs to the NNs are again normalized to $[0, 1]$, and we use perturbation bounds 0.016, 0.02, and 0.024, which correspond to roughly 4, 5, and 6 pixel values.

We consider 4 different linear over-approximations, including the existing linear bounds described in Sec. 3.1 (lin) and tangent planes to the non-linear bounds as discussed in Sec. 5. We consider 3 pairs of tangent planes corresponding to the different nonlinear bounds proposed in this work: ER is derived from L^{ER} and U^{ER}; LSE is derived from L^{LSE} and U^{LSE}; LSE$*$ is derived from L^{LSE} and U^{LSE}. In all cases we take the tangent plane passing through the midpoint of the softmax input range ($\frac{l+u}{2}$).

Table 4: % of Instances Certified by Different Softmax Bounds for MNIST classifiers

Net. (Acc.)	**Pert.**	lin	ER	LSE	LSE
`A_small` (89.2)	4/256	74.0	83.6	79.4	**84.0**
	5/256	67.8	79.2	73.8	**81.2**
	6/256	61.2	74.6	68.6	**76.4**
`A_med` (98.2)	4/256	60.0	81.2	84.4	**84.6**
	5/256	30.0	62.4	69.2	**71.6**
	6/256	11.0	34.2	39.4	**46.0**
`A_big` (99)	4/256	42.0	65.0	68.6	**70.8**
	5/256	13.0	29.4	29.6	**41.2**
	6/256	1.6	6.2	3.6	**11.6**

The percentages of verified instances (out of the 500 test images) using different linear bounds are shown in Table 4. Overall, the new linearized bounds (last 3 columns) result in significantly higher verification precision than lin does. We believe this is because lin is obtained by linearization at each step of the decomposition, which results in higher accumulation of approximation errors. Among the new linearized bounds, while LSE$*$ consistently certifies more instances than ER, LSE and ER are evenly matched with head-to-head wins for both. Upon further examination, we discover that while LSE is less precise overall, it certifies 9 instances that LSE$*$ is unable to solve and 77 instances that ER is unable to solve. Overall, the 4 methods combined certify 2628 of the 4500 instances, in contrast to 2572 instances certified by LSE$*$ alone. This suggests the benefit of a portfolio approach. Rules for deciding which linearized bounds to use are also an interesting future direction.

SST In addition, we perform the same robustness verification task on an NLP transformer model trained on the SST-2 dataset (Socher et al., 2013). SST-2 is a sentiment analysis dataset consisting of movie reviews, where each review is labeled as either positive or negative. In this setting, perturbation is performed on the embedding of the input sentence. The trained transformer obtains 74% natural accuracy and the robust accuracies verified by different configurations are shown in Table 5. The results further confirm the benefit of the newly proposed linearized bounds (last three columns) over the existing linear bounds (lin). They also highlight the complementary nature of the proposed bounds as unlike in Table 4, LSE is superior to LSE$*$.

Table 5: % of Instances Certified by Different Softmax Bounds for the SST-2 transformer

Pert. (ℓ_∞)	lin	ER	LSE	LSE
0.02	**68.8**	**68.8**	**68.8**	**68.8**
0.04	**63.2**	**63.2**	**63.2**	**63.2**
0.06	56.6	56.8	**57.6**	57.4
0.08	51.4	52.0	**52.4**	**52.4**
0.1	45.0	46.2	**47.2**	46.8
0.12	39.0	39.8	**40.6**	**40.6**
0.14	31.8	33.0	**34.0**	33.6
0.16	26.4	26.8	**28.2**	27.2

8 CONCLUSION

We have provided convex bounds on the softmax function satisfying the hierarchy in (1), both theoretically and numerically, and used them in certifying the robustness of uncertainty estimators and transformers. Future work could consider the development of more customized algorithms for solving the resulting convex optimization problems, and/or further exploitation of the linearized bounds in Section 5 to facilitate scaling to larger networks. We also hope for an even better lower bound from the LSE approach, one that might be provably stronger than L^{ER} for $K > 2$ or more smoothly integrate the two bounds L^{LSE}, L^{LSE} .

Artifact

Scripts to reproduce the experiments in Sections 6 and 7 can be found at `https://github.com/NeuralNetworkVerification/bounding-softmax/tree/aistats`.

Acknowledgements

This work was partially supported by IBM as a founding member of Stanford Institute for Human-centered Artificial Intelligence (HAI). Additional support was provided by a direct grant from HAI and by the Stanford Center for AI Safety.

References

Greg Anderson, Shankara Pailoor, Isil Dillig, and Swarat Chaudhuri. Optimization and abstraction: A synergistic approach for analyzing neural network robustness. In *Proc. Programming Language Design and Implementation (PLDI)*, page 731–744, 2019.

Stanley Bak, Hoang-Dung Tran, Kerianne Hobbs, and Taylor T Johnson. Improved geometric path enumeration for verifying relu neural networks. In *International Conference on Computer Aided Verification*, pages 66–96. Springer, 2020.

Leonard Berrada, Sumanth Dathathri, Krishnamurthy Dvijotham, Robert Stanforth, Rudy R Bunel, Jonathan Uesato, Sven Gowal, and M. Pawan Kumar. Make sure you're unsure: A framework for verifying probabilistic specifications. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 11136–11147, 2021. URL https://proceedings.neurips.cc/paper/2021/file/5c5bc7df3d37b2a7ea29e1b47b2bd4ab-Paper.pdf.

Julian Bitterwolf, Alexander Meinke, and Matthias Hein. Certifiably adversarially robust detection of out-of-distribution data. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 16085–16095, 2020. URL https://proceedings.neurips.cc/paper/2020/file/b90c46963248e6d7aab1e0f429743ca0-Paper.pdf.

Gregory Bonaert, Dimitar I. Dimitrov, Maximilian Baader, and Martin Vechev. Fast and precise certification of transformers. In *Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation (PLDI)*, PLDI 2021, page 466–481, 2021. URL https://doi.org/10.1145/3453483.3454056.

Akhilan Boopathy, Tsui-Wei Weng, Pin-Yu Chen, Sijia Liu, and Luca Daniel. Cnn-cert: An efficient framework for certifying robustness of convolutional neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3240–3247, 2019.

Elena Botoeva, Panagiotis Kouvaros, Jan Kronqvist, Alessio Lomuscio, and Ruth Misener. Efficient verification of relu-based neural networks via dependency analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3291–3299, 2020.

Guillaume Bouchard. Efficient bounds for the softmax function and applications to approximate inference in hybrid models. In *NeurIPS Workshop for Approximate Bayesian Inference in Continuous/Hybrid Systems*, 2007. URL https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.331.8075&rep=rep1&type=pdf.

Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Rudy Bunel, Jingyue Lu, Ilker Turkaslan, Pushmeet Kohli, P Torr, and P Mudigonda. Branch and bound for piecewise linear neural network verification. *Journal of Machine Learning Research*, 21(2020), 2020.

Steven Diamond and Stephen Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

Souradeep Dutta, Susmit Jha, Sriram Sankaranarayanan, and Ashish Tiwari. Output range analysis for deep feedforward neural networks. In *NASA Formal Methods - 10th International Symposium, NFM 2018, Newport News, VA, USA, April 17-19, 2018, Proceedings*, 2018.

Krishnamurthy Dvijotham, Robert Stanforth, Sven Gowal, Timothy A Mann, and Pushmeet Kohli. A dual approach to scalable verification of deep networks. In *UAI*, volume 1, page 3, 2018.

Ruediger Ehlers. Formal verification of piece-wise linear feed-forward neural networks. In *International Symposium on Automated Technology for Verification and Analysis*, pages 269–286. Springer, 2017.

Aymeric Fromherz, Klas Leino, Matt Fredrikson, Bryan Parno, and Corina Păsăreanu. Fast geometric projections for local robustness certification. *arXiv preprint arXiv:2002.04742*, 2020.

Timon Gehr, Matthew Mirman, Dana Drachsler-Cohen, Petar Tsankov, Swarat Chaudhuri, and Martin T. Vechev. AI2: safety and robustness certification of neural networks with abstract interpretation. In *2018 IEEE Symposium on Security and Privacy, SP 2018, Proceedings, 21-23 May 2018, San Francisco, California, USA*, pages 3–18, 2018. doi: 10.1109/SP.2018.00058. URL https://doi.org/10.1109/SP.2018.00058.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1243–1252, 2017.

Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007. URL https://doi.org/10.1198/016214506000001437.

Sven Gowal, Krishnamurthy Dj Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Relja Arandjelovic, Timothy Mann, and Pushmeet Kohli. Scalable verified training for provably robust image

classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4842–4851, 2019.

Patrick Henriksen and Alessio Lomuscio. Deepsplit: An efficient splitting method for neural network verification via indirect effect analysis. In *Proceedings of the 30th international joint conference on artificial intelligence (IJCAI21). To Appear. ijcai. org*, 2021.

Xiaowei Huang, Marta Kwiatkowska, Sen Wang, and Min Wu. Safety verification of deep neural networks. In *CAV*, 2017.

Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer. Reluplex: An efficient smt solver for verifying deep neural networks. In *International Conference on Computer Aided Verification*, pages 97–117. Springer, 2017.

Guy Katz, Derek A Huang, Duligur Ibeling, Kyle Julian, Christopher Lazarus, Rachel Lim, Parth Shah, Shantanu Thakoor, Haoze Wu, Aleksandar Zeljić, et al. The marabou framework for verification and analysis of deep neural networks. In *International Conference on Computer Aided Verification*, pages 443–452, 2019.

Haitham Khedr, James Ferlez, and Yasser Shoukry. Peregrinn: Penalized-relaxation greedy neural network verifier. *arXiv preprint arXiv:2006.10864*, 2020.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017. URL https://proceedings.neurips.cc/paper/2017/file/9ef2ed4b7fd2c810847ffa5fa85bce38-Paper.pdf.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, 2015.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

Frank Nielsen and Ke Sun. Guaranteed bounds on information-theoretic measures of univariate mixtures using piecewise log-sum-exp inequalities. *Entropy*, 18(12), 2016. URL https://www.mdpi.com/1099-4300/18/12/442.

Brendan O'Donoghue, Eric Chu, Neal Parikh, and Stephen Boyd. SCS: Splitting conic solver, version 3.2.1. https://github.com/cvxgrp/scs, November 2021.

Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Semidefinite relaxations for certifying robustness to adversarial examples. *arXiv preprint arXiv:1811.01057*, 2018.

Rahul Rahaman and Alexandre Thiery. Uncertainty quantification and deep ensembles. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 20063–20075, 2021. URL https://proceedings.neurips.cc/paper/2021/file/a70dc40477bc2adceef4d2c90f47eb82-Paper.pdf.

Hadi Salman, Greg Yang, Huan Zhang, Cho-Jui Hsieh, and Pengchuan Zhang. A convex relaxation barrier to tight robustness verification of neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/246a3c5544feb054f3ea718f61adfa16-Paper.pdf.

Zhouxing Shi, Huan Zhang, Kai-Wei Chang, Minlie Huang, and Cho-Jui Hsieh. Robustness verification for transformers. In *International Conference on Learning Representations (ICLR)*, 2020. URL https://openreview.net/forum?id=BJxwPJHFwS.

Gagandeep Singh, Timon Gehr, Matthew Mirman, Markus Püschel, and Martin Vechev. Fast and effective robustness certification. *Advances in Neural Information Processing Systems*, 31:10802–10813, 2018a.

Gagandeep Singh, Markus Püschel, and Martin T. Vechev. A practical construction for decomposing numerical abstract domains. *Proc. ACM Program. Lang.*, 2(POPL): 55:1–55:28, 2018b.

Gagandeep Singh, Rupanshu Ganvir, Markus Püschel, and Martin Vechev. Beyond the single neuron convex barrier for neural network certification. *Advances in Neural Information Processing Systems*, 32:15098–15109, 2019a.

Gagandeep Singh, Timon Gehr, Markus Püschel, and Martin Vechev. An abstract domain for certifying neural networks. *Proceedings of the ACM on Programming Languages*, 3(POPL):1–30, 2019b.

Gagandeep Singh, Timon Gehr, Markus Püschel, and Martin Vechev. Boosting robustness certification of neural networks. In *International Conference on Learning Representations*, 2019c.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

Michalis K. Titsias. One-vs-each approximation to softmax for scalable estimation of probabilities. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 29, 2016. URL https://proceedings.neurips.cc/paper/2016/file/814a9c18f5abff398787c9cfcbf3d80c-Paper.pdf.

Christian Tjandraatmadja, Ross Anderson, Joey Huchette, Will Ma, KRUNAL KISHOR PATEL, and Juan Pablo Vielma. The convex relaxation barrier, revisited: Tightened single-neuron relaxations for neural network verification. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 21675–21686. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/f6c2a0c4b566bc99d596e58638e342b0-Paper.pdf.

Vincent Tjeng, Kai Y. Xiao, and Russ Tedrake. Evaluating robustness of neural networks with mixed integer programming. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=HyGIdiRqtm.

Hoang-Dung Tran, Stanley Bak, Weiming Xiang, and Taylor T Johnson. Verification of deep convolutional neural networks using imagestars. In *International Conference on Computer Aided Verification*, pages 18–42. Springer, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. volume 30, 2017.

Joseph A Vincent and Mac Schwager. Reachable polyhedral marching (rpm): A safety verification algorithm for robotic systems with deep neural network components. *arXiv preprint arXiv:2011.11609*, 2020.

Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang, and Suman Jana. Formal security analysis of neural networks using symbolic intervals. In *27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018*, pages 1599–1614, 2018. URL https://www.usenix.org/conference/usenixsecurity18/presentation/wang-shiqi.

Shiqi Wang, Huan Zhang, Kaidi Xu, Xue Lin, Suman Jana, Cho-Jui Hsieh, and J Zico Kolter. Beta-crown: Efficient bound propagation with per-neuron split constraints for complete and incomplete neural network verification. *arXiv preprint arXiv:2103.06624*, 2021.

Lily Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Luca Daniel, Duane Boning, and Inderjit Dhillon. Towards fast computation of certified robustness for relu networks. In *International Conference on Machine Learning*, pages 5276–5285. PMLR, 2018a.

Tsui-Wei Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Duane Boning, Inderjit S Dhillon, and Luca Daniel. Towards fast computation of certified robustness for relu networks. *International Coference on International Conference on Machine Learning*, 2018b.

Eric Wong and Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, pages 5286–5295, 2018.

Haoze Wu, Aleksandar Zeljić, Guy Katz, and Clark Barrett. Efficient neural network analysis with sum-of-infeasibilities. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 143–163. Springer, 2022.

Min Wu, Matthew Wicker, Wenjie Ruan, Xiaowei Huang, and Marta Kwiatkowska. A game-based approximate verification of deep neural networks with provable guarantees. *Theoretical Computer Science*, 807:298–329, 2020.

Weiming Xiang, Hoang-Dung Tran, and Taylor T Johnson. Output reachable set estimation and verification for multilayer neural networks. *IEEE transactions on neural networks and learning systems*, 29(11):5777–5783, 2018.

Kaidi Xu, Huan Zhang, Shiqi Wang, Yihan Wang, Suman Jana, Xue Lin, and Cho-Jui Hsieh. Fast and complete: Enabling complete neural network verification with rapid and massively parallel incomplete verifiers. *arXiv preprint arXiv:2011.13824*, 2020.

Tom Zelazny, Haoze Wu, Clark Barrett, and Guy Katz. On optimizing back-substitution methods for neural network verification. *arXiv preprint arXiv:2208.07669*, 2022.

Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh, and Luca Daniel. Efficient neural network robustness certification with general activation functions. *Advances in neural information processing systems*, 31, 2018.

A RELATED WORK ON NEURAL NETWORK VERIFICATION

Researchers have proposed several techniques for verifying properties of neural networks (Katz et al., 2017; Singh et al., 2019b; Ehlers, 2017; Gehr et al., 2018; Tjeng et al., 2019; Bunel et al., 2020; Wang et al., 2018). To overcome the inherent scalability limitations, state-of-the-art verifiers seek a good balance between scalability and precision, by designing customized abstractions Tran et al. (2020); Ehlers (2017); Huang et al. (2017); Wu et al. (2020); Singh et al. (2019b,a); Xiang et al. (2018); Gehr et al. (2018), bound-propagation passes Zelazny et al. (2022); Zhang et al. (2018); Dutta et al. (2018); Tjeng et al. (2019); Weng et al. (2018a); Singh et al. (2018a), or convex optimization procedures Salman et al. (2019); Tjandraatmadja et al. (2020); Raghunathan et al. (2018); Wang et al. (2021); Singh et al. (2019c); Boopathy et al. (2019); Wu et al. (2022). These abstraction-based methods have been integrated into case-analysis-based search shell to ensure completeness (Katz et al., 2017, 2019; Ehlers, 2017; Bak et al., 2020; Tran et al., 2020; Vincent and Schwager, 2020; Henriksen and Lomuscio, 2021; Fromherz et al., 2020; Anderson et al., 2019; Tjeng et al., 2019; Bunel et al., 2020; Khedr et al., 2020; Botoeva et al., 2020; Xu et al., 2020; Wu et al., 2022). Our convex optimization procedure could be integrated in a search shell to obtain more precise over-approximation of the output sets. The linear bounds that we propose can also be integrated into sub-polyhedral abstraction domains (Singh et al., 2018b) other than DeepPoly/CROWN.

B NOTATION

Table 6 summarizes the more important symbols used in the main paper.

C PROOFS

C.1 Proof of Theorem 1

Proof. The inequality $L^{\text{lin}}(\tilde{x}) \leq L^{\text{ER}}(\tilde{x})$ is a consequence of applying the first inequality in (11) to (12a) (with $q_1 = \overline{\text{SE}}(\tilde{x}; \tilde{l}, \tilde{u})$) to yield (16a).

To establish the inequality $U^{\text{ER}}(\tilde{x}) \leq U^{\text{lin}}(\tilde{x})$, we first recognize that

$$\underline{q}_1^{\text{lin}} = 1 + \sum_{j=2}^{K} e^{t_j}(\tilde{l}_j - t_j + 1) \leq \text{SE}(\tilde{l}) = \frac{1}{\overline{p}_1}$$

using (7a). Hence

$$\begin{aligned} U^{\text{ER}}(\tilde{x}) &= \overline{p}_1 \underbrace{\left(1 - \underline{p}_1 \,\text{SE}(\tilde{x})\right)}_{0 \text{ for } \tilde{x} \in [\tilde{l}, \tilde{u}]} + \underline{p}_1 \\ &\leq \frac{1}{\underline{q}_1^{\text{lin}}} \left(1 - \underline{p}_1 \,\text{SE}(\tilde{x})\right) + \underline{p}_1 \\ &\leq \frac{1}{\underline{q}_1^{\text{lin}}} \left(1 - \underline{p}_1 \left(1 + \sum_{j=2}^{K} e^{t_j}(\tilde{x}_j - t_j + 1)\right)\right) + \underline{p}_1 \\ &= U^{\text{lin}}(\tilde{x}), \end{aligned}$$

where the second inequality is again due to (7a). □

C.2 Proof of Theorem 2

Proof. To prove the inequality for the **upper bounds**, we rewrite $U^{\text{ER}}(x)$ and $U^{\text{LSE}}(x)$ as convex combinations of the constant bounds $\underline{p}_1$ and $\overline{p}_1$. We then compare the coefficients in the two convex combinations.

We rewrite $U^{\text{ER}}(x)$ (16b) as a convex combination as follows, making use of the identities $\underline{p}_1 \,\text{SE}(\tilde{u}) = \overline{p}_1 \,\text{SE}(\tilde{l}) = 1$ from

Table 6: Summary of Notation

symbol	description
x_j	jth input to softmax or network
p_j	jth output of softmax (probability)
K	number of softmax inputs/outputs
$\tilde{x}_j$	difference $x_j - x_1$
$\dot{x}_j$	difference $x_j - x_{j^*}$, where $j \ = \arg\max_j(l_j + u_j)$ (defined below)
l_j, u_j	lower and upper bounds on x_j
m_j	midpoint $(l_j + u_j)/2$ of range of x_j
ϵ_j	half-width $(u_j - l_j)/2$ of range of x_j (same for all j if subscript j omitted)
$\tilde{l}_j, \tilde{u}_j$	lower and upper bounds on $\tilde{x}_j$
$\dot{l}_j, \dot{u}_j$	lower and upper bounds on $\dot{x}_j$
$\underline{p}_1, \overline{p}_1$	constant lower and upper bounds on p_j (5)
$L_j(x), U_j(x)$	lower and upper bounds on p_j (p_1 if subscript j omitted) as functions of x
$L^{\mathrm{lin}}(x), U^{\mathrm{lin}}(x)$	linear lower and upper bounds on p_1 (12) combining Shi et al. (2020); Bonaert et al. (2021)
$L^{\mathrm{ER}}(x), U^{\mathrm{ER}}(x)$	new nonlinear lower and upper bounds on p_1 (16) from exponential-reciprocal (ER) decomposition
$L^{\mathrm{LSE}}(x)$	first nonlinear lower bound on p_1 (23a) from log-sum-exp (LSE) decomposition
$L^{\mathrm{LSE}}\ (x)$	second nonlinear lower bound on p_1 (23b) from log-sum-exp (LSE) decomposition
$L^{\mathrm{LSE}_2}(x)$	nonlinear lower bound on p_1 for $K = 2$ (24) from log-sum-exp (LSE) decomposition
$U^{\mathrm{LSE}}(x)$	nonlinear upper bound on p_1 (23c) from log-sum-exp (LSE) decomposition
$\mathrm{SE}(x)$	sum-of-exponentials functions $\sum_{j=1}^K e^{x_j}$
$\overline{\mathrm{SE}}(x; l, u)$	chordal upper bound on $\mathrm{SE}(x)$ (9) parametrized by l, u
$\mathrm{LSE}(x)$	log-sum-exp function $\log\left(\sum_{j=1}^K e^{x_j}\right)$
q_1	intermediate variable in ER decomposition of p_1
$\underline{q}_1, \overline{q}_1$	constant lower and upper bounds on q_1 (10)
t_j, t_{q_1}	tangent points used in the linear bounds $L^{\mathrm{lin}}(x)$, $U^{\mathrm{lin}}(x)$
r_1	intermediate variable in LSE decomposition of p_1
x	clean input
y	ground truth label
$S(p, y\ )$	scoring rule for evaluating p
superscript m	index of model in deep ensemble

(5) in the second and fourth lines below:

$$
\begin{aligned}
U^{\mathrm{ER}}(x) &= \overline{p}_1 + \underline{p}_1 - \overline{p}_1\underline{p}_1\,\mathrm{SE}(\tilde{x})\frac{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{l})}{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{l})} \\
&= \overline{p}_1\left(1 - \frac{\mathrm{SE}(\tilde{x})}{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{l})}\right) + \underline{p}_1\left(1 + \frac{\mathrm{SE}(\tilde{x})}{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{l})}\right) \\
&= \overline{p}_1\frac{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{l}) - \mathrm{SE}(\tilde{x})}{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{l})} + \underline{p}_1\frac{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{l}) + \mathrm{SE}(\tilde{x})}{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{l})} \\
&= \overline{p}_1\frac{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{x})}{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{l})} - \frac{1}{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{l})} + \underline{p}_1\frac{\mathrm{SE}(\tilde{x}) - \mathrm{SE}(\tilde{l})}{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{l})} + \frac{1}{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{l})} \\
&= \overline{p}_1\frac{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{x})}{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{l})} + \underline{p}_1\frac{\mathrm{SE}(\tilde{x}) - \mathrm{SE}(\tilde{l})}{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{l})}.
\end{aligned} \tag{31}
$$

The two fractions above are non-negative and sum to 1, so this is indeed a convex combination.

For $U^{\mathrm{LSE}}(x)$ (23c), we use the fact that $-\log(\underline{p}_1) = \mathrm{LSE}(\tilde{u})$ and $-\log(\overline{p}_1) = \mathrm{LSE}(\tilde{l})$ to bring it closer to the expression

in (31):

$$U^{\mathrm{LSE}}(x) = \overline{p}_1 \frac{-\log(\underline{p}_1) - \mathrm{LSE}(\tilde{x})}{\log(\overline{p}_1) - \log(\underline{p}_1)} + \underline{p}_1 \frac{\mathrm{LSE}(\tilde{x}) + \log(\overline{p}_1)}{\log(\overline{p}_1) - \log(\underline{p}_1)}$$
$$= \overline{p}_1 \frac{\mathrm{LSE}(\tilde{u}) - \mathrm{LSE}(\tilde{x})}{\mathrm{LSE}(\tilde{u}) - \mathrm{LSE}(\tilde{l})} + \underline{p}_1 \frac{\mathrm{LSE}(\tilde{x}) - \mathrm{LSE}(\tilde{l})}{\mathrm{LSE}(\tilde{u}) - \mathrm{LSE}(\tilde{l})}. \tag{32}$$

Again the two fractions above are non-negative and sum to 1.

To show that $U^{\mathrm{LSE}}(x) \leq U^{\mathrm{ER}}(x)$, it now suffices to show that the coefficient of $\overline{p}_1$ in (31) is greater than or equal to the coefficient of $\overline{p}_1$ in (32) (equivalently, one could compare the $\underline{p}_1$ coefficients). This can be done using the concavity of the logarithm function as follows:

$$\begin{aligned}
\mathrm{LSE}(\tilde{x}) &= \log(\mathrm{SE}(\tilde{x})) \\
&\geq \frac{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{x})}{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{l})} \log(\mathrm{SE}(\tilde{l})) + \frac{\mathrm{SE}(\tilde{x}) - \mathrm{SE}(\tilde{l})}{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{l})} \log(\mathrm{SE}(\tilde{u})) \\
&= \frac{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{x})}{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{l})} \mathrm{LSE}(\tilde{l}) + \left(1 - \frac{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{x})}{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{l})}\right) \mathrm{LSE}(\tilde{u}).
\end{aligned}$$

The above can be rearranged to yield

$$\frac{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{x})}{\mathrm{SE}(\tilde{u}) - \mathrm{SE}(\tilde{l})} \geq \frac{\mathrm{LSE}(\tilde{u}) - \mathrm{LSE}(\tilde{x})}{\mathrm{LSE}(\tilde{u}) - \mathrm{LSE}(\tilde{l})},$$

thus completing the proof for the upper bounds.

To prove the inequality for the **lower bounds** in the case $K = 2$, we rewrite $L^{\mathrm{ER}}(x)$ (16a) as

$$\begin{aligned}
L^{\mathrm{ER}}(x) &= \left(1 + \frac{\tilde{u}_2 - \tilde{x}_2}{\tilde{u}_2 - \tilde{l}_2} e^{\tilde{l}_2} + \frac{\tilde{x}_2 - \tilde{l}_2}{\tilde{u}_2 - \tilde{l}_2} e^{\tilde{u}_2}\right)^{1} \\
&= \left(\frac{\tilde{u}_2 - \tilde{x}_2}{\tilde{u}_2 - \tilde{l}_2} \left(1 + e^{\tilde{l}_2}\right) + \frac{\tilde{x}_2 - \tilde{l}_2}{\tilde{u}_2 - \tilde{l}_2} \left(1 + e^{\tilde{u}_2}\right)\right)^{1} \\
&= \left(\frac{\tilde{u}_2 - \tilde{x}_2}{\tilde{u}_2 - \tilde{l}_2} \left(\overline{p}_1\right)^{1} + \frac{\tilde{x}_2 - \tilde{l}_2}{\tilde{u}_2 - \tilde{l}_2} \left(\underline{p}_1\right)^{1}\right)^{1},
\end{aligned}$$

using (5) to obtain the last line. This last line can be recognized as a weighted harmonic mean of the constant bounds $\overline{p}_1$ and $\underline{p}_1$, with weights $(\tilde{u}_2 - \tilde{x}_2)/(\tilde{u}_2 - \tilde{l}_2)$ and $(\tilde{x}_2 - \tilde{l}_2)/(\tilde{u}_2 - \tilde{l}_2)$. On the other hand, for $K = 2$, $L^{\mathrm{LSE}_2}(x)$ (24) is a weighted geometric mean of the same quantities with the same weights. It follows from the inequality of (weighted) harmonic and geometric means[4] that $L^{\mathrm{ER}}(x) \geq L^{\mathrm{LSE}_2}(x)$. □

D ALTERNATIVE LOG-SUM-EXP LOWER BOUND FOR $K > 2$

This appendix describes a third lower bound arising from the log-sum-exp decomposition of Section 4, as an alternative to (21), (22).

Our motivation is to generalize inequality (19), which applies when $K = 2$ and leads to a provably tighter overall bound (24) than $L^{\mathrm{ER}}(x)$ (16a) (Theorem 2). We start by rewriting $-\mathrm{LSE}(\tilde{x})$ as

$$-\mathrm{LSE}(\tilde{x}) = -\log\left(1 + e^{\ x_1} \sum_{j=2}^{K} e^{x_j}\right) = -\log\left(1 + e^{\mathrm{LSE}(x_2^K)\ \ x_1}\right), \tag{33}$$

[4] This can be proven as a corollary of the arithmetic mean-geometric mean inequality, among other ways.

where $x_2^K = (x_2, \ldots, x_K)$. We then apply inequality (19) to (33) with $\mathrm{LSE}(x_2^K) - x_1$ in place of $\tilde{x}_2$. To do this, we also have to replace $\tilde{l}_2$, $\tilde{u}_2$ with lower and upper bounds on $\mathrm{LSE}(x_2^K) - x_1$. Given the monotonicity of $\mathrm{LSE}(x_2^K)$, we use the bounds

$$\underline{v}_1 = \mathrm{LSE}\left(l_2^K\right) - u_1, \tag{34a}$$

$$\overline{v}_1 = \mathrm{LSE}\left(u_2^K\right) - l_1. \tag{34b}$$

Then the application of (19) to (33) yields

$$-\mathrm{LSE}(\tilde{x}) \geq \underbrace{-\frac{\log(\overline{p}_1) - \log(\underline{p}_1)}{\overline{v}_1 - \underline{v}_1}}_{0} \left(\mathrm{LSE}\left(x_2^K\right) - x_1\right) + \frac{\overline{v}_1 \log(\overline{p}_1) - \underline{v}_1 \log(\underline{p}_1)}{\overline{v}_1 - \underline{v}_1}, \tag{35}$$

where (5) has been used to identify $-\log(1 + e^{\underline{v}_1}) = \log(\overline{p}_1)$ and $-\log(1 + e^{\overline{v}_1}) = \log(\underline{p}_1)$.

For $K = 2$, the lower bound in (35) is affine in x and hence convex, but for $K > 2$, it is concave because of the negative multiplier in front. To address the non-convexity, we further bound $\mathrm{LSE}(x_2^K)$ using chordal bounds similar to (7b):

$$\mathrm{LSE}\left(x_2^K\right) = \log\left(\sum_{j=2}^{K} e^{x_j}\right) \leq \log\left(\sum_{j=2}^{K}\left(\frac{u_j - x_j}{u_j - l_j} e^{l_j} + \frac{x_j - l_j}{u_j - l_j} e^{u_j}\right)\right). \tag{36}$$

Substituting (36) into (35) gives

$$-\mathrm{LSE}(\tilde{x}) \geq \frac{\overline{v}_1 \log(\overline{p}_1) - \underline{v}_1 \log(\underline{p}_1)}{\overline{v}_1 - \underline{v}_1} + \frac{\log(\overline{p}_1) - \log(\underline{p}_1)}{\overline{v}_1 - \underline{v}_1}\left(x_1 - \log\left(\sum_{j=2}^{K}\left(\frac{u_j - x_j}{u_j - l_j} e^{l_j} + \frac{x_j - l_j}{u_j - l_j} e^{u_j}\right)\right)\right). \tag{37}$$

Since the term after x_1 is the composition of an affine function of x_2^K with $-\log$, the right-hand side of (37) is now a convex function of x.

An overall lower bound on the softmax function is obtained by exponentiating (37):

$$L^{\mathrm{LSE}'}(x) = \exp\left[\frac{\overline{v}_1 \log(\overline{p}_1) - \underline{v}_1 \log(\underline{p}_1)}{\overline{v}_1 - \underline{v}_1} + \frac{\log(\overline{p}_1) - \log(\underline{p}_1)}{\overline{v}_1 - \underline{v}_1}\left(x_1 - \log\left(\sum_{j=2}^{K}\left(\frac{u_j - x_j}{u_j - l_j} e^{l_j} + \frac{x_j - l_j}{u_j - l_j} e^{u_j}\right)\right)\right)\right]. \tag{38}$$

In the additional synthetic experiment results reported in Appendix F, we do not see a regime in which $L^{\mathrm{LSE}'}$ is better than the larger of L^{LSE}, L^{LSE} , i.e., no regime in which $L^{\mathrm{LSE}'}$ is the uniquely best lower bound. We did not include $L^{\mathrm{LSE}'}$ in the main paper for this reason.

E LINEARIZED BOUNDS

We here present the partial derivatives of the non-linear lower- and upper- bounds presented in the paper, in addition to those of L^{ER}.

$\frac{\partial L^{\mathrm{LSE}}(x)}{\partial x_i}$ can be computed by applying the product rule and the chain rule. The results are:

$$\frac{\partial L^{\mathrm{LSE}}(x)}{\partial x_1} = L^{\mathrm{LSE}}(x) - e^{x_1} s(x)^2 \left(\frac{e^{u_i} - e^{l_i}}{u_i - l_i}\right) \tag{39}$$

$$\frac{\partial L^{\mathrm{LSE}}(x)}{\partial x_i} = -e^{x_1} s(x)^2 \left(\frac{e^{u_i} - e^{l_i}}{u_i - l_i}\right) \quad \text{for } i \neq 1 \tag{40}$$

where $s(x) = 1/\overline{\mathrm{SE}}(x; l, u)$ from (23a).

$\frac{\partial L^{\mathrm{LSE}*}(x)}{\partial x_i}$ is slightly more complicated. When $j = 1$, $\frac{\partial L^{\mathrm{LSE}*}(x)}{\partial x_i}$ is the same as $\frac{\partial L^{\mathrm{ER}}(x)}{\partial x_i}$. In the case where $j \neq 1$, $\frac{\partial L^{\mathrm{LSE}*}(x)}{\partial x_i}$ can be again computed with applications of chain rules. The results are:

$$\frac{\partial L^{\mathrm{LSE}}(x)}{\partial x_1} = L^{\mathrm{LSE}}(x) - \frac{e^{\dot{x}_1}}{\overline{\mathrm{SE}}(\dot{x},\dot{l},\dot{u})^2}\frac{e^{\dot{u}_1} - e^{\dot{l}_1}}{\dot{u}_1 - \dot{l}_1} \tag{41}$$

$$\frac{\partial L^{\mathrm{LSE}}(x)}{\partial x_{j^*}} = -L^{\mathrm{LSE}}(x) + \frac{e^{\dot{x}_1}}{\overline{\mathrm{SE}}(\dot{x},\dot{l},\dot{u})^2} \cdot \sum_{i \neq j^*}^{K} \left(\frac{e^{\dot{u}_i} - e^{\dot{l}_i}}{\dot{u}_i - \dot{l}_i} \right) \tag{42}$$

$$\frac{\partial L^{\mathrm{LSE}}(x)}{\partial x_i} = -\frac{e^{\dot{x}_1}}{\overline{\mathrm{SE}}(\dot{x},\dot{l},\dot{u})^2}\frac{e^{\dot{u}_i} - e^{\dot{l}_i}}{\dot{u}_i - \dot{l}_i} \quad \text{for } i \notin \{1, j\} \tag{43}$$

$\frac{\partial U^{\mathrm{ER}}(x)}{\partial x_i}$ can be computed with the chain rule. The results are:

$$\frac{\partial U^{\mathrm{ER}}(x)}{\partial x_1} = \overline{p}_1 \underline{p}_1 \left(\mathrm{SE}(\tilde{x}) - 1\right) \tag{44}$$

$$\frac{\partial U^{\mathrm{ER}}(x)}{\partial x_i} = -\overline{p}_1 \underline{p}_1 e^{\tilde{x}_i} \quad \text{for } i \neq 1 \tag{45}$$

The partial derivative $\frac{\partial U^{\mathrm{LSE}}(x)}{\partial x_i}$ can be computed by rewriting $\mathrm{LSE}(\tilde{x})$ as $\mathrm{LSE}(x) - x_1$, and applying the chain rule and the fact that $\frac{\partial\,\mathrm{LSE}(x)}{\partial x_i} = \frac{e^{x_i}}{\mathrm{SE}(x)}$. The results are:

$$\frac{\partial U^{\mathrm{LSE}}(x)}{\partial x_1} = -\frac{\overline{p}_1 - \underline{p}_1}{\log(\overline{p}_1) - \log(\underline{p}_1)} \left(\frac{e^{x_1}}{\mathrm{SE}(x)} - 1 \right) \tag{46}$$

$$\frac{\partial U^{\mathrm{LSE}}(x)}{\partial x_i} = -\frac{\overline{p}_1 - \underline{p}_1}{\log(\overline{p}_1) - \log(\underline{p}_1)} \cdot \frac{e^{x_i}}{\mathrm{SE}(x)} \quad \text{for } i \neq 1 \tag{47}$$

F SYNTHETIC DATA EVALUATION: DETAILS AND VARIATIONS

This appendix contains additional material on the synthetic experiment in Section 6: details on data generation, variations of Figure 2 in the main paper, and an explanation of the looseness of the U^{lin} bound.

F.1 Data Generation Details

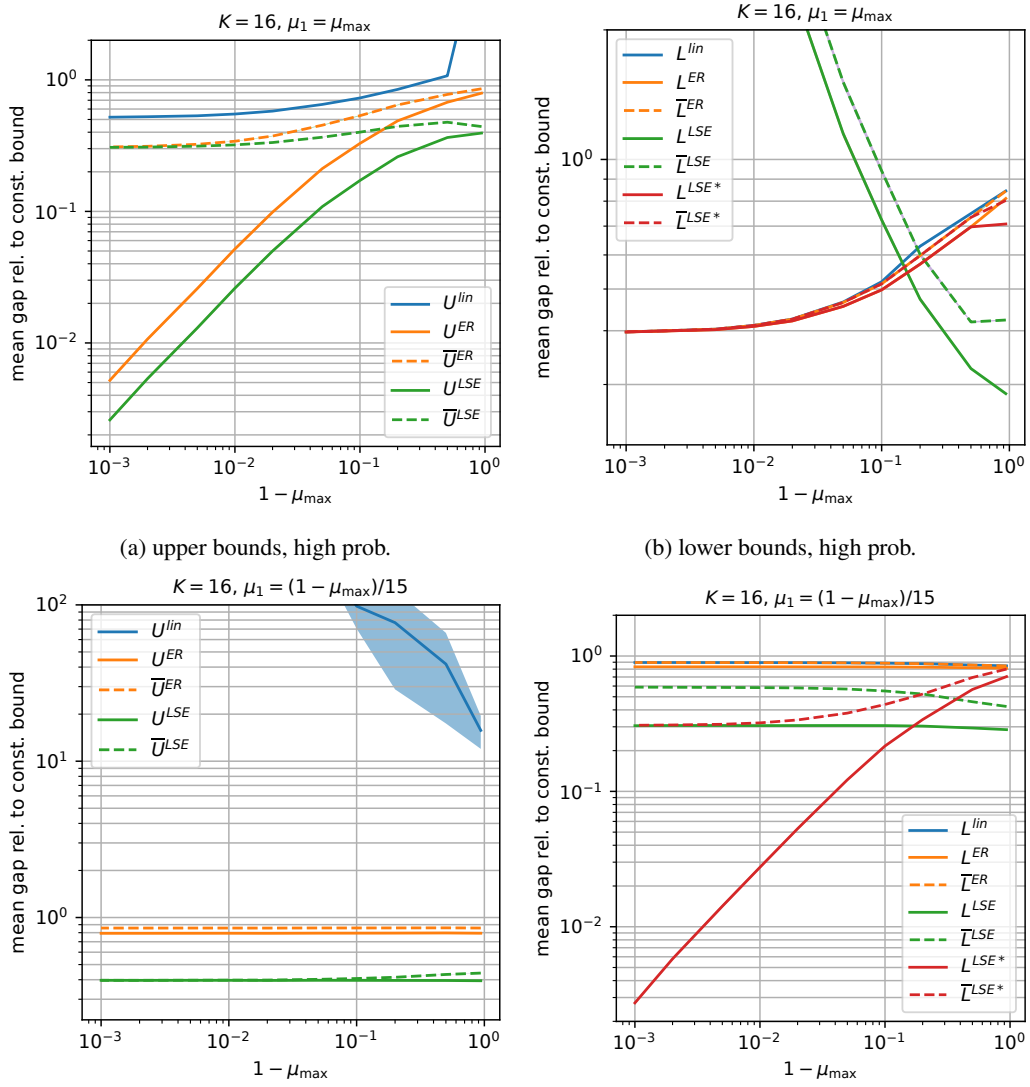
Recall that we sample softmax outputs from a Dirichlet distribution and choose one component of the Dirichlet mean, $\mu_{j_{\max}}$, to be larger than the others, $\mu_{j_{\max}} = \mu_{\max} \geq \mu_j$ for $j \neq j_{\max}$. This is done by setting the Dirichlet concentration parameters to be $\alpha_{j_{\max}} = \alpha_{\max} \geq 1$ and $\alpha_j = 1$ for $j \neq j_{\max}$. Then the largest mean component is given by

$$\mu_{\max} = \frac{\alpha_{\max}}{\sum_{j=1}^{K} \alpha_j} = \frac{\alpha_{\max}}{\alpha_{\max} + K - 1}.$$

After sampling a softmax output p, we convert it to an input m (i.e., logits) by taking $m_j = \log(p_j/p_1)$ and then centering m by subtracting the mean of the m_j's.

F.2 Variations on Figure 2

Linearized Bounds In Figure 3, we add the linearized bounds of Section 5 to Figures 2a, 2c, 2e, 2g. The linearized bounds are plotted using dashed lines of the same color as their non-linearized counterparts. In Figures 3b, 3c (lower bounds on a probability with high mean and upper bounds on a low probability), the losses in strength due to linearization are modest. However in Figures 3a, 3d, the mean gap ratios of the linearized bounds (relative to the respective constant bound) appear to be limited to no lower than 0.3, whereas the mean gap ratios of U^{ER}, U^{LSE}, and L^{LSE} decrease to much smaller values. Thus the gain due to nonlinearity appears to be substantial in these two scenarios (upper bound on high probability and lower bound on low probability).



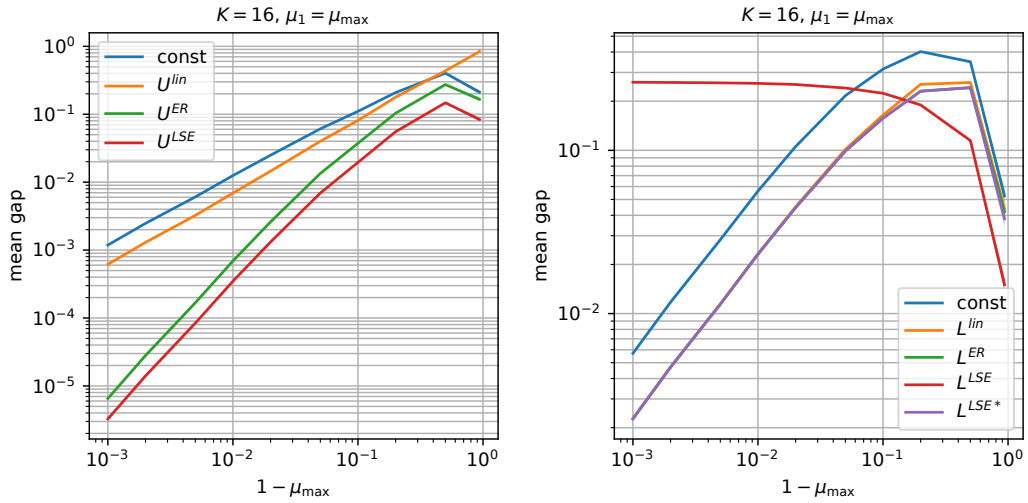
(a) upper bounds, high prob.

(b) lower bounds, high prob.

(c) upper bounds, low prob.

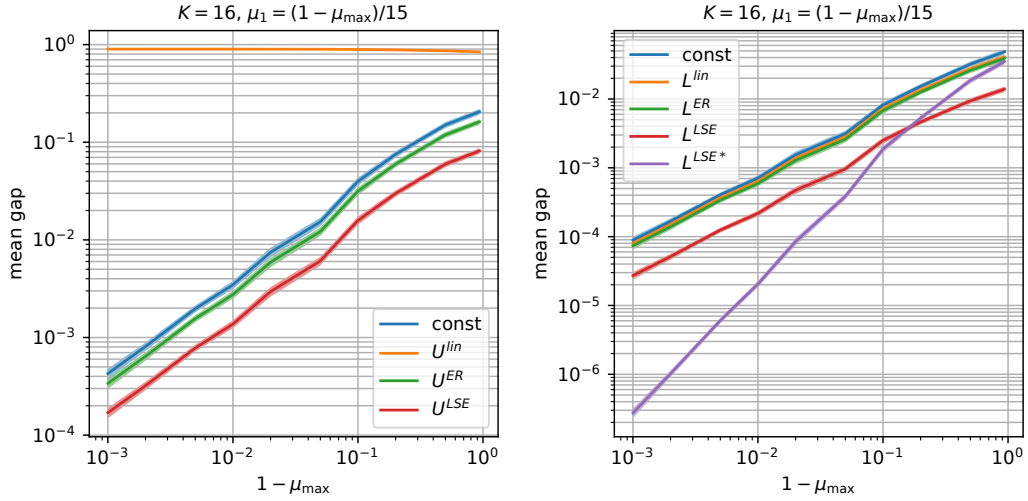
(d) lower bounds, low prob.

Figure 3: Mean gaps of upper bounds (left column) and lower bounds (right column) on softmax output p_1, now including linearized bounds (dashed lines), for synthetically generated input regions of width $\epsilon = 1$. In the top/bottom row, the mean μ_1 of p_1 is high/low.



(a) upper bounds, high prob.

(b) lower bounds, high prob.



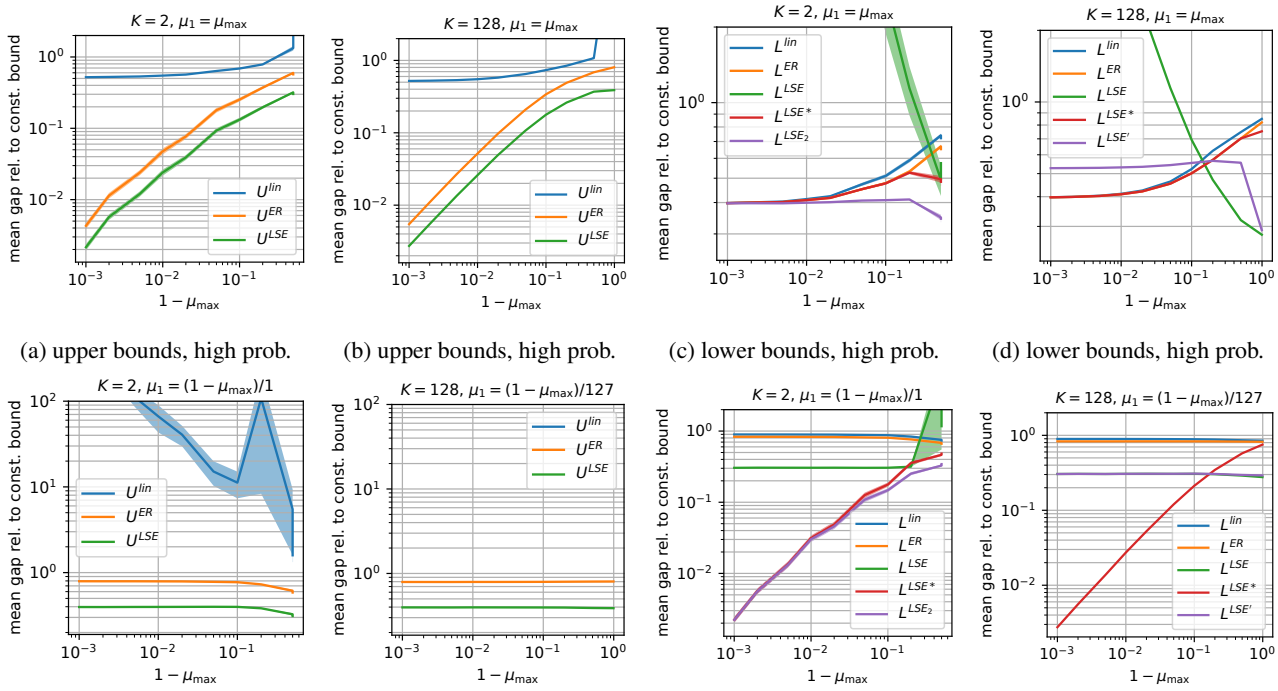
(c) upper bounds, low prob.

(d) lower bounds, low prob.

Figure 4: Mean gaps of upper bounds (left column) and lower bounds (right column) on softmax output p_1 for synthetically generated input regions of width $\epsilon = 1$. Unlike in Figure 2, no ratios of mean gaps are taken. In the top/bottom row, the mean μ_1 of p_1 is high/low.

Mean Gaps Without Taking Ratios Figure 4 is another version of Figures 2a, 2c, 2e, 2g in which the mean gaps themselves are plotted, without dividing by the mean gap of the constant bound which is now plotted separately. The general pattern is that the mean gaps decrease as $\mu_{\max} \to 1$ ($\mu_1 \to 1$ in the top row of Figure 4, $\mu_1 \to 0$ in the bottom row). In Figures 4a, 4b, the mean gaps also tend to decrease as $\mu_{\max} \to 0$. The two exceptions are U^{lin} in Figure 4c, which is uniformly poor across the $\mu_{\max}$ range, and L^{LSE} in Figure 4b, which is the best lower bound for smaller $\mu_{\max}$ but deterioriates at higher $\mu_{\max}$. These are the same behaviors seen in Figures 2c, 2e.

Different Values of K Figure 5 shows versions of Figures 2a, 2c, 2e, 2g with dimensions $K = 2$ and $K = 128$ instead of $K = 16$. In Figure 5f, U^{lin} is worse than the constant bound $\overline{p}_1$ by more than a factor of 100 (the upper limit of the plot is kept at 100 for consistency). For $K = 2$ in Figures 5c, 5g, L^{LSE_2} refers to bound (24), which is provably tighter than L^{ER}. Figures 5c, 5g show that L^{LSE_2} is superior to L^{LSE}, L^{LSE} as well over the entire range of $\mu_{\max}$. For $K = 128$ in Figures 5d, 5h, $L^{\text{LSE}'}$ is the bound (38) derived in Appendix D. As claimed earlier, $L^{\text{LSE}'}$ does not improve upon the better of L^{LSE}, L^{LSE} (in Figure 5h, the curve for L^{LSE} largely coincides with that for $L^{\text{LSE}'}$ but the former is slightly lower as $\mu_{\max} \to 0$). Aside from the additions of L^{LSE_2} and $L^{\text{LSE}'}$, the patterns for the other bounds are similar to those in Figure 2.

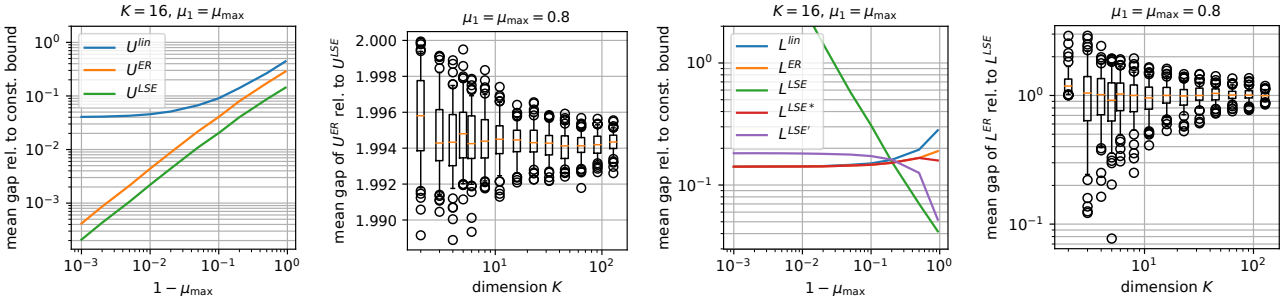


(a) upper bounds, high prob. (b) upper bounds, high prob. (c) lower bounds, high prob. (d) lower bounds, high prob.

(e) upper bounds, low prob. (f) upper bounds, low prob. (g) lower bounds, low prob. (h) lower bounds, low prob.

Figure 5: Mean gaps of upper bounds (left two columns) and lower bounds (right two columns) on softmax output p_1 for synthetically generated input regions of width $\epsilon = 1$ and dimensions $K = 2$ and $K = 128$. In the top/bottom row, the mean μ_1 of p_1 is high/low. For $K = 2$ in Figures 5c, 5g, we use (24) for the L^{LSE} curve.

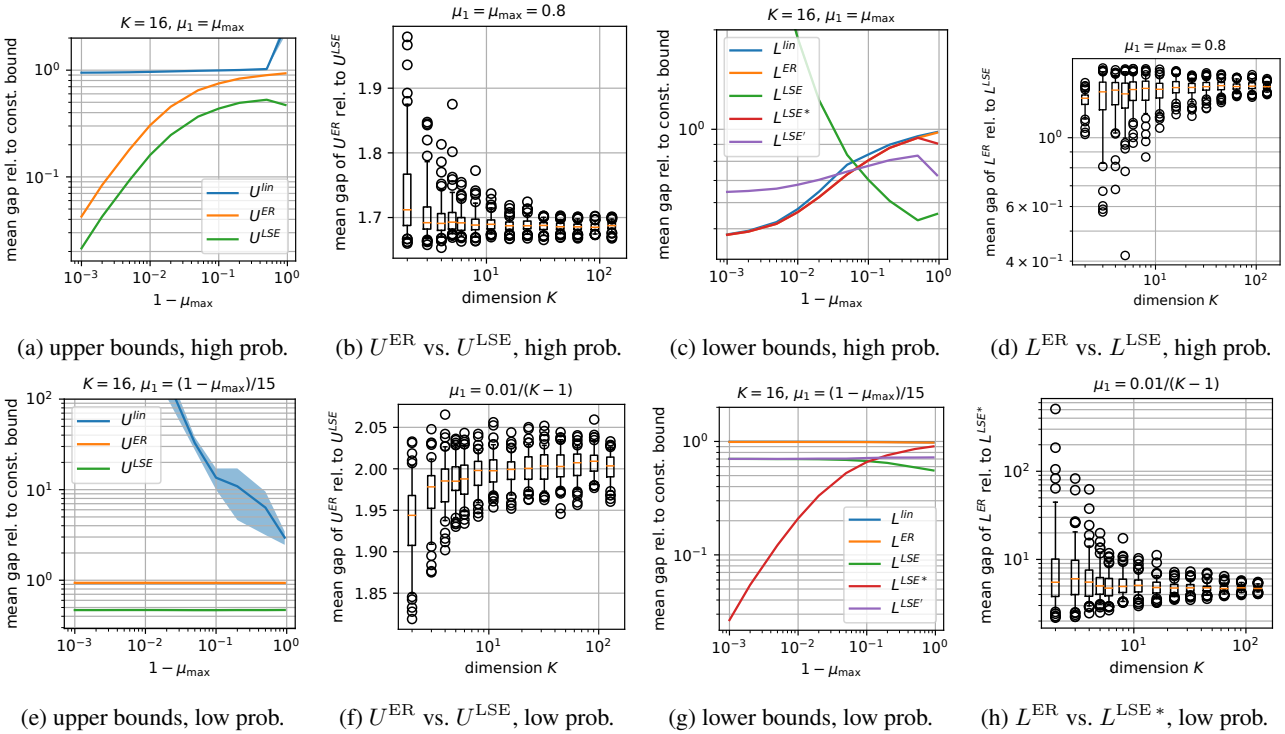
Different Values of ϵ Figures 6 and 7 are versions of Figure 2 with input region width $\epsilon = 0.2$ and $\epsilon = 2$ respectively instead of $\epsilon = 1$. The most notable difference is that for $\epsilon = 0.2$ in Figure 6, the problem of bounding softmax is easier and the mean gap ratios are generally lower (i.e., improvement over the constant bounds is greater). In particular in Figures 6a, 6e, U^{lin} is not excessively loose and does improve upon the constant bound $\overline{p}_1$.



(a) upper bounds, high prob. (b) U^{ER} vs. U^{LSE}, high prob. (c) lower bounds, high prob. (d) L^{ER} vs. L^{LSE}, high prob.

(e) upper bounds, low prob. (f) U^{ER} vs. U^{LSE}, low prob. (g) lower bounds, low prob. (h) L^{ER} vs. $L^{\mathrm{LSE}*}$, low prob.

Figure 6: Mean gaps of upper bounds (left two columns) and lower bounds (right two columns) on softmax output p_1 for synthetically generated input regions of width $\epsilon = 0.2$. In the top/bottom row, the mean μ_1 of p_1 is high/low.



(a) upper bounds, high prob. (b) U^{ER} vs. U^{LSE}, high prob. (c) lower bounds, high prob. (d) L^{ER} vs. L^{LSE}, high prob.

(e) upper bounds, low prob. (f) U^{ER} vs. U^{LSE}, low prob. (g) lower bounds, low prob. (h) L^{ER} vs. $L^{\mathrm{LSE}*}$, low prob.

Figure 7: Mean gaps of upper bounds (left two columns) and lower bounds (right two columns) on softmax output p_1 for synthetically generated input regions of width $\epsilon = 2$. In the top/bottom row, the mean μ_1 of p_1 is high/low.

F.3 Looseness of U^{lin}

We provide an explanation of the looseness of the linear ER bound $U^{\mathrm{lin}}(x)$ (12b) (perhaps not the only explanation).

Recall that the quantity t_j appearing in (12b) is the tangent point (8) chosen to bound the exponential $e^{\tilde{x}_j}$ from below by a tangent line. The second term in (8) ensures that the tangent line is non-negative for $\tilde{x}_j \in [\tilde{l}_j, \tilde{u}_j]$. Since $e^{\tilde{x}_j}$ is a highly nonlinear function, for large enough intervals $[\tilde{l}_j, \tilde{u}_j]$ it is likely that we need to set $t_j = \tilde{l}_j + 1$ for non-negativity. Suppose then that $t_j = \tilde{l}_j + 1$ for all $j = 2, \ldots, K$. Then substitution into (10a) yields $\underline{q}_1^{\mathrm{lin}} = 1$, and substitution into (12b) gives

$$U^{\mathrm{lin}}(\tilde{x}) = 1 - \underline{p}_1 \sum_{j=2}^{K} e^{\tilde{l}_j+1}(\tilde{x}_j - \tilde{l}_j). \tag{48}$$

When $\tilde{x} = \tilde{l}$, (48) implies $U^{\mathrm{lin}}(\tilde{x}) = 1$, which is a trivial upper bound. We can also ask when (48) is better (i.e., smaller) than the constant bound $\overline{p}_1$. This occurs when

$$\sum_{j=2}^{K} e^{\tilde{l}_j+1}(\tilde{x}_j - \tilde{l}_j) \geq \frac{1 - \overline{p}_1}{\underline{p}_1}, \tag{49}$$

i.e., when $\tilde{x}$ is large enough compared to $\tilde{l}$ for the above inequality to hold.

The above explanation is consistent with numerical results in Figures 2a, 2e, 6a, 6e, 7a, 7e. When $\epsilon = 0.2$ in Figures 6a, 6e, the intervals $[\tilde{l}_j, \tilde{u}_j]$ are smaller and U^{lin} is not as poor as when ϵ and $[\tilde{l}_j, \tilde{u}_j]$ are larger and the situation above occurs more frequently. In addition, U^{lin} is worse compared to $\overline{p}_1$ in the low μ_1 setting of Figures 2e, 6e, 7e than in the high μ_1 setting. In this case, both $\underline{p}_1$ and $\overline{p}_1$ tend to be small, the right-hand side of (49) is large, and (49) may not hold for any $\tilde{x} \in [\tilde{l}, \tilde{u}]$.

G FULL FORMULATION OF UNCERTAINTY ESTIMATION SCORE MAXIMIZATION

We consider an ensemble of M feedforward NNs, each consisting of L hidden layers with ReLU activations. Let $x^{\ell,m}$ denote the neurons in layer ℓ of network m, $\ell = 0, \ldots, L$, $m = 1, \ldots, M$, before affine transformation is applied. The input to the ensemble corresponds to $\ell = 0$ and is the same for all networks: $x^{0,m} = x^0$. Let $W^{\ell,m}$ and $b^{\ell,m}$ be the weights and biases of the affine transformation in layer ℓ of network m, and $z^{\ell,m}$ be the output of the affine transformation. $z^{L,m}$ is the set of logits from network m and is the input to a softmax function with probabilities p^m as output. The output probabilities of the ensemble are the averages of the network probabilities, $p = (1/M)\sum_m p^m$. We assume that we have lower and upper bounds $l^{\ell,m}$, $u^{\ell,m}$ on each $z^{\ell,m}$. In our experiment, these are obtained using the CROWN/DeepPoly (Zhang et al., 2018; Singh et al., 2019b) abstract interpretation.

Below we give the full set of constraints in the score maximization problem (27) for verification of uncertainty estimation.

Input We consider ℓ_1 perturbations up to radius ϵ of the given input x (i.e., $\mathcal{B}_1(x, \epsilon)$ in (27)). This can be expressed as the following linear constraints on x^0:

$$x - \epsilon \leq x^0 \leq x + \epsilon. \tag{50}$$

Hidden Layers Given bounds $l^{\ell,m}$, $u^{\ell,m}$ on the pre-activation neurons $z^{\ell,m}$ for layers $l = 0, \ldots, L-1$, we can partition the post-activation neurons $x^{\ell+1,m}$ into three sets $\mathcal{I}, \mathcal{A}, \mathcal{U}$ (we may regard all components of x^0 as belonging to $\mathcal{A}$):

1. *Inactive*, $\mathcal{I} = \{j : u_j^{\ell,m} \leq 0\}$: In this case, $x_j^{\ell+1,m} = 0$ and can be dropped as an input to the next affine transformation.
2. *Active*, $\mathcal{A} = \{j : l_j^{\ell,m} \geq 0\}$: This implies that $x_j^{\ell+1,m} = z_j^{\ell,m}$ and $x_j^{\ell+1,m}$ is an affine function of $x^{\ell,m}$. With $x_{\mathcal{A}}^{\ell+1,m}$ denoting the subvector of $x^{\ell+1,m}$ indexed by $\mathcal{A}$ and using similar notation for other vectors and matrices, these affine functions can be written as

$$x_{\mathcal{A}}^{\ell+1,m} = W_{\mathcal{A},\mathcal{A}\cup\mathcal{U}}^{\ell,m} x_{\mathcal{A}\cup\mathcal{U}}^{\ell,m} + b_{\mathcal{A}}^{\ell,m}, \qquad \ell = 0, \ldots, L-1,\ m = 1, \ldots, M. \tag{51}$$

3. *Unstable*, $\mathcal{U} = \{j : l_j^{\ell,m} < 0 < u_j^{\ell,m}\}$: Here the ReLU function remains nonlinear and we approximate it using the triangular linear relaxation of Ehlers (2017). This gives us the following constraints:

$$z_{\mathcal{U}}^{\ell,m} = W_{\mathcal{U},\mathcal{A}[\mathcal{U}}^{\ell,m} x_{\mathcal{A}[\mathcal{U}}^{\ell,m} + b_{\mathcal{U}}^{\ell,m}, \tag{52}$$

$$x_{\mathcal{U}}^{\ell+1,m} \geq z_{\mathcal{U}}^{\ell,m}, \qquad x_{\mathcal{U}}^{\ell+1,m} \geq 0, \tag{53}$$

$$x_j^{\ell+1,m} \leq \frac{u_j^{\ell,m}}{u_j^{\ell,m} - l_j^{\ell,m}} \left(z_j^{\ell,m} - l_j^{\ell,m} \right), \quad j \in \mathcal{U},\ \ell = 0, \ldots, L-1,\ m = 1, \ldots, M. \tag{54}$$

Logits These are given by affine transformation of the last hidden layer:

$$z^{L,m} = W_{,\mathcal{A}[\mathcal{U}}^{L,m} x_{\mathcal{A}[\mathcal{U}}^{L,m} + b^{L,m}. \tag{55}$$

Probabilities We impose the simplex constraint in (6) on the probabilities from each network:

$$\sum_{k=1}^{K} p_k^m = 1, \qquad p_k^m \geq 0, \qquad k = 1, \ldots, K,\ m = 1, \ldots, M. \tag{56}$$

Objective Function Equation (28) is used for the NLL scoring rule and (30) for Brier score, with $p = (1/M) \sum_{m=1}^{M} p^m$ in both cases. The fact that (30) is an upper bound on the Brier score can be seen from the chordal upper bound on the function p_k^2 over the interval $[\underline{p}_k, \overline{p}_k]$. This chordal upper bound can be simplified to

$$p_k^2 \leq (\underline{p}_k + \overline{p}_k) p_k - \underline{p}_k \overline{p}_k.$$

Softmax We use the bounds developed in this work to relate logits to probabilities:

$$p_k^m \geq L_k \left(z^{L,m} \right), \tag{57a}$$

$$p_k^m \leq U_k \left(z^{L,m} \right), \tag{57b}$$

where the subscript k in L_k, U_k refers to the bound for the kth softmax output (all the bounds presented in the paper were for $k = 1$). In our experiments, $L_k \in \{L^{\text{lin}}, L^{\text{ER}}, L^{\text{LSE}}\}$ and $U_k \in \{U^{\text{lin}}, U^{\text{LSE}}\}$ as discussed in Section 7.1. However, because of the form of the objective functions (28), (30), only one of (57a), (57b) is needed for each k. Specifically, for $k = y$, the probability p_{y^*} is minimized in both objective functions[5], whereas for $k \neq y$, p_k is maximized[6]. Therefore only (57a) is used for $k = y$ and (57b) for $k \neq y$.

In addition, we also impose the constant bounds

$$p_k^m \geq \underline{p}_k^m, \tag{58a}$$

$$p_k^m \leq \overline{p}_k^m, \tag{58b}$$

where again only (58a) is used for $k = y$ and (58b) for $k \neq y$. In the case of the linear pair $(L^{\text{lin}}, U^{\text{lin}})$, constraints (58) help considerably because U^{lin} in particular is sometimes not better than $\overline{p}_k^m$ (recall for example Figure 2e). For the nonlinear pairs $(L^{\text{ER}}, U^{\text{LSE}})$, $(L^{\text{LSE}}, U^{\text{LSE}})$, (58) can be helpful in improving the numerical precision of the optimal objective value.

Summary In summary, problem (27) is subject to constraints (50)–(58), where (57a), (58a) are used for $k = y$ and (57b), (58b) for $k \neq y$.

H DETAILS ON MODEL ARCHITECTURES AND TRAINING

We describe the details of the deep ensembles deployed in predictive uncertainty estimation of Section 7.1 as well as the self-attention models used in Section 7.2.

[5] In the case of Brier score (30), the coefficient of p_{y^*} is $(\ 2 + \underline{p}_k + \overline{p}_k)\ \ 0$ and hence p_{y^*} is minimized as in (28).

[6] In the case of NLL, the maximization is implicit because p_{y^*} is minimized and (56) couples p_{y^*} to the other p_k's.

H.1 Deep Ensembles

We follow the layer structures of the deep ensemble models for the MNIST dataset as in Section 3.4 of Lakshminarayanan et al. (2017). While the authors did not opensource the accompanying implementation codes for this paper, we are advised that the uncertainty-baselines[7] Python package can be utilised to construct the training pipeline. In particular, we look into the ./baselines/mnist/ directory and train the following MNIST deep ensemble models in Table 7. Specifically, we have two ensembles, MNIST and MNIST-large, each containing 5 networks with identical layer structure yet different weights and biases after the training process. Overall, networks of the MNIST ensemble reach $\sim 94.5\%$ test accuracy, while the MNIST-large ensemble networks achieve $\sim 97.95\%$. Table 1 in Section 7.1 shows verification results for MNIST, while results for MNIST-large are in Table 12 in Appendix I.2. We mention that, as recommended by Lakshminarayanan et al. (2017), the MNIST-large models are adversarially trained using the Projected Gradient Descent (Madry et al., 2018) attacking method from the Adversarial Robustness Toolbox[8] Python library. Similarly, we adversarially train an ensemble comprising 5 networks on the CIFAR-10 dataset, and report their verification results in Table 2 of Section 7.1. The structure of these CIFAR-10 models is in Table 8, and their test accuracy varies from 28.49% to 42.71%.

Table 7: Architecture for the MNIST (left) and MNIST-large (right) deep ensemble models.

Layer Type	Parameter	Activation
Input	$28 \times 28 \times 1$	–
Flatten	784	–
Fully Connected	784×10	ReLU
Fully Connected	10×10	ReLU
Fully Connected	10×10	Softmax

Layer Type	Parameter	Activation
Input	$28 \times 28 \times 1$	–
Flatten	784	–
Fully Connected	784×100	ReLU
Fully Connected	100×100	ReLU
Fully Connected	100×100	ReLU
Fully Connected	100×10	Softmax

Table 8: Architecture for the CIFAR-10 deep ensemble model.

Layer Type	Parameter	Activation
Input	$32 \times 32 \times 3$	–
Flatten	3072	–
Fully Connected	3072×20	ReLU
Fully Connected	20×20	ReLU
Fully Connected	20×20	ReLU
Fully Connected	20×10	Softmax

H.2 Self-Attention Models

We adapt the self-attention mechanism proposed in Vaswani et al. (2017), where essentially encoders and decoders comprising attention blocks are used in processing sentences, into classifying the MNIST and SST-2 datasets. The layer structure of our MNIST self-attention model is outlined in Table 9. Specifically, we train 3 models with increasing dimensions of the self-attention block in terms of the number of attention heads and the size of each head: `A_small`, `A_med`, `A_big` with test accuracy 90.25%, 97.41%, and 98.28%, respectively. Similarly, we deploy adversarial training by using the Projected Gradient Descent (Madry et al., 2018) attacking method from the Adversarial Robustness Toolbox Python library. As for the SST-2 sentiment analysis dataset, we adversarially train the self-attention model in Table 10. While the self-attention block is the same as `A_small` for MNIST, it has the embedding layer to accommodate text inputs and the sigmoid function of the last layer to produce sentiment, i.e., either positive or negative. The test accuracy of this model is 75.34%.

[7] Access via GitHub repository `https://github.com/google/uncertainty-baselines`

[8] Access via GitHub repository `https://github.com/Trusted-AI/adversarial-robustness-toolbox`

Table 9: Architecture for the MNIST self-attention models: `A_small` (90.25%), `A_med` (97.41%), and `A_big` (98.28%).

Layer Type	Parameter Size	Activation
Input	$28 \times 28 \times 1$	–
Flatten	–	–
Fully Connected	`A_small`: 16; `A_med`: 144; `A_big`: 256	ReLU
Reshape	`A_small`: $(2, 8)$; `A_med`: $(4, 36)$; `A_big`: $(4, 64)$	–
MultiHeadAttention	`A_small`: num_heads=2, key_dim=4 `A_med`: num_heads=3, key_dim=12 `A_big`: num_heads=4, key_dim=16	–
Flatten	–	–
Fully Connected	10	Softmax

Table 10: Architecture for the SST-2 self-attention model (75.34%).

Layer Type	Parameter Size	Activation
Input	50	–
Embedding	vocab_size=4000, embedding_dim=16, input_length=50	–
Flatten	–	–
Fully Connected	16	ReLU
Reshape	$(2, 8)$	–
MultiHeadAttention	num_heads=2, key_dim=4	–
Flatten	–	–
Fully Connected	1	Sigmoid

I ADDITIONAL UNCERTAINTY ESTIMATION RESULTS

I.1 Lower Bound on UQ Scores Computed by PGD Attack

We perform projected gradient descent (PGD) attack to obtain a lower bound of the UQ scores within the given perturbation bound. The results are shown in Table 11.

Table 11: Upper Bounds on Uncertainty Estimation Scores Using Different Softmax Bounds

Score (Clean)	ϵ	PGD	$L^{\text{lin}}, U^{\text{lin}}$	$L^{\text{ER}}, U^{\text{LSE}}$	$L^{\text{LSE}}, U^{\text{LSE}}$
NLL (0.105)	2/256	0.202	0.265	0.261	**0.251**
	3/256	0.285	0.442	0.433	**0.420**
	4/256	0.407	0.726	0.697	**0.690**
Brier (0.048)	2/256	0.061	0.138	0.134	**0.131**
	3/256	0.073	0.244	0.235	**0.234**
	4/256	0.104	0.417	**0.403**	**0.403**

I.2 Verification of a Larger Deep Ensemble

Table 12 shows upper bounds on expected uncertainty estimation scores for the MNIST-large ensemble in the same manner as Table 1 for MNIST in Section 7.1. The table also includes results for two larger perturbation radii, 5/256 and 6/256 (0.020 and 0.024 in $[0, 1]$-normalized units). In the case of MNIST-large, the lower bound L^{LSE} appears to coincide with

L^{ER} for all 100 test instances, and thus the two rightmost columns in Table 12 are identical. Nevertheless, there is still consistent improvement in going from linear to nonlinear bounds.

Table 12: Upper Bounds on Uncertainty Estimation Scores for a Larger Deep Ensemble Using Different Softmax Bounds

Score (Clean)	ϵ	PGD	$L^{\mathrm{lin}}, U^{\mathrm{lin}}$	$L^{\mathrm{ER}}, U^{\mathrm{LSE}}$	$L^{\mathrm{LSE}}, U^{\mathrm{LSE}}$
NLL (0.0150)	2/256	0.0196	0.0212	0.0210	0.0210
	3/256	0.0222	0.0263	0.0257	0.0257
	4/256	0.0252	0.0339	0.0326	0.0326
	5/256	0.0286	0.0449	0.0430	0.0430
	6/256	0.0325	0.0611	0.0585	0.0585
Brier (0.00373)	2/256	0.00584	0.00677	0.00659	0.00659
	3/256	0.00707	0.00961	0.00921	0.00921
	4/256	0.00847	0.01418	0.01350	0.01350
	5/256	0.00976	0.02161	0.02065	0.02065
	6/256	0.01125	0.03336	0.03206	0.03206

I.3 UQ Scores Computed by Considering Each Network Separately

Since the output of the ensemble models we study are average of the outputs of the individual models, another way to compute the UQ scores for an ensemble models is to computing the scores for each individual model and take the average. Note that this is not equivalent to analyzing the ensemble model as a whole, because the constraint that each model always take the same input is relaxed and the resulting average score is an over-approximation. One benefit though is that analyzing each individual model is computationally more efficient than analyzing the ensemble model as a whole.

Table 13: Upper Bounds on Uncertainty Estimation Scores Using Different Softmax Bounds

Score (Clean)	ϵ	$L^{\mathrm{lin}}, U^{\mathrm{lin}}$	$L^{\mathrm{ER}}, U^{\mathrm{LSE}}$	$L^{\mathrm{LSE}}, U^{\mathrm{LSE}}$
NLL (0.105)	2/256	0.270	**0.263**	**0.263**
	3/256	0.448	**0.435**	0.436
	4/256	0.732	**0.712**	**0.712**
Brier (0.048)	2/256	0.139	**0.136**	**0.136**
	3/256	0.245	**0.240**	**0.240**
	4/256	0.418	**0.412**	**0.412**

Table 14: Average runtime in seconds

Dataset	$L^{\mathrm{lin}}, U^{\mathrm{lin}}$	$L^{\mathrm{ER}}, U^{\mathrm{LSE}}$	$L^{\mathrm{LSE}}, U^{\mathrm{LSE}}$
Combined	10.9	91.6	92.4
Separate	3.36	43.22	42.49