Abstract

This paper develops a certified defense for deep neural network (DNN) based content based image retrieval (CBIR) against adversarial examples (AXs). Previous works put their effort into certified defense for classification to improve certified robustness, which guarantees that no AX to cause misclassification exists around the sample. Such certified defense, however, could not be applied to CBIR directly because the goals of adversarial attack against classification and CBIR are completely different. To develop the certified defense for CBIR, we first define new certified robustness of CBIR, which guarantees that no AX that changes the ranking of CBIR exists around the sample. Such certified defense, however, could not be applied to CBIR directly because the goals of adversarial attack against classification and CBIR are completely different. To develop the certified defense for CBIR, we first define new certified robustness of CBIR, which guarantees that no AX that changes the ranking of CBIR exists around the query or candidate images. Then, we propose computationally tractable verification algorithms that verify whether the certified robustness of CBIR is achieved by utilizing upper and lower bounds of distances between feature representations of perturbed and non-perturbed images. Finally, we propose new objective functions for training feature extraction DNNs that increases the number of inputs that satisfy the certified robustness of CBIR by tightening the upper and lower bounds. Experimental results show that our objective functions significantly improve the certified robustness of CBIR than existing methods.

1. Introduction

Content based image retrieval (CBIR) is a task that retrieves visually similar images to a given query image from a set of candidate images. Modern CBIR performs retrieval by ranking the similarity between the query image and candidate images based on feature extraction deep neural networks (DNNs) trained by metric learning [4, 3]. However, recent studies reveal that such DNN-based CBIR is vulnerable to small human-imperceptible perturbation to the input data, called adversarial examples (AXs) [38, 40, 12, 22, 26, 39, 14, 28, 1]. Such AXs can be input to DNN-based CBIR as query or candidate images and maliciously modify the ranking results by manipulating the output of the feature extraction DNNs. Since the DNN-based CBIR is often involved in security-critical systems such as face identification [16] and person re-identification [34], defense methods for DNN-based CBIR against AXs are necessary.

A great deal of effort has been devoted to empirical defense methodologies for the classification task [15, 33, 17]. Adversarial training [15], which trains DNNs using AXs as training data, is one of the most effective empirical defense methodologies for the classification. Adversarial training has also been shown to be effective in CBIR empirically [38, 40]. While these empirical defense methods achieve robustness against conventional attacks, they often suffer from adaptive attacks [23], which assume the attacker is aware of the strategy of the defense method. Since there is no guarantee that these empirical defense methods are effective against adaptive attacks, defense methods that achieve robustness against AXs with theoretical guarantees are needed to deal with adaptive attacks.

To overcome adaptive attacks, many studies have worked to establish defense with certified robustness of classification. Certified robustness means that there is no AX to cause misclassification within an \(l_p\)-ball centered on a given sample. This type of defense is generally referred to as certified defense. Certified defense generally consists of (i) a verification algorithm to verify whether a given classifier satisfies certified robustness at a given sample and (ii) robust training for classifier to increase the number of samples that satisfy certified robustness. Since exactly verifying whether a given classifier satisfies certified robustness at a given sample is known to be reduced to an NP-complete
problem [8, 29], [6, 7, 35, 30, 36, 21, 24, 37, 31] make the problem relaxed and computationally tractable. Precisely, they use the upper and lower bounds of logits against AXs in the \( l_p \)-ball instead of exact logits for the verification. Using the bounds makes the verification computationally tractable, while the results can include false negative, i.e., given samples are determined to be not robust, even when they actually achieve certified robustness (conversely, samples determined to be non-robust are guaranteed to be always non-robust). Considering that this gap is caused by the looseness of the bounds, robust training to make this bound tighter has been introduced [6, 7, 35, 30, 36, 21, 24]. By training the DNN in this way, we can expect to reduce the number of cases where samples that are robust are judged to be non-robust.

Although certified defense for classification has been investigated extensively, no attention has been paid to certified defense for CBIR. Moreover, the existing certified defense for classification cannot be directly applied to CBIR in the following sense: the goals of adversarial attack against classification and CBIR are completely different. Specifically, the adversarial attacks against classification aim to change the predicted class label of the classifier, whereas the adversarial attacks against CBIR aim to change the rank of the similarity between the query image and candidate images calculated by feature extraction DNNs. To realize certified defense for CBIR, we need to introduce a definition specifically designed for certified robustness of CBIR. Then, we must design verification algorithms to verify whether a given feature extraction DNN satisfies the new robustness at given inputs in computationally tractable ways and robustness training of feature extraction DNNs suitable for the new verification algorithm.

1.1. Our Contributions

In this paper, we develop a certified defense for CBIR. Our contribution is three-fold. First, we define new certified robustness of CBIR. Our certified robustness means that, given a feature extraction DNN, query image, and candidate images, there is a guarantee that no AX that changes the ranking of CBIR exists within \( l_\infty \)-balls centered on the query or candidate images.

Second, we propose computationally tractable verification algorithms for the certified robustness of CBIR. To exactly verify whether a given feature extraction DNN satisfies our certified robustness at given query and candidate images, we need to evaluate the exact maximum and minimum distance between AXs in \( l_p \)-balls centered on the query image and benign candidate images in the feature space (or AXs in \( l_p \)-balls centered on the candidate images and benign query images). That makes the verification computationally intractable. To alleviate this, our algorithms use upper and lower bounds of the distances obtained by applying interval bound propagation (IBP) [6, 7, 35] to feature extraction DNNs.

Third, we propose new objective functions to train feature extraction DNNs that attain tighter evaluation of the upper and lower bound of the distances. When the bounds are loose, our verification algorithms can judge inputs as non-robust. To decrease such misjudging, we propose to train DNN with a regularization term that encourages the bounds on the distances to be tighter.

We experimentally show that our objective functions significantly improve the certified robustness compared to existing methods, including adversarial training for CBIR [40] and robust training for improving certified robustness of classification task [6]. To the best of our knowledge, this is the first study that achieves certified defense for CBIR.

2. Preliminaries

2.1. Content Based Image Retrieval (CBIR)

CBIR is a task to find images similar to a query image in a set of candidate images. Let \( X \) be the instance space. Let \( q \in X \) be a query image and \( C = \{c_i | c_i \in X\}_{i=1}^N \) be a set of candidate images where \( N \) is the number of candidate images. Let \( f : X \rightarrow \mathbb{R}^d \) be a feature extractor where \( d \) is the feature dimension. Then, CBIR ranks \( c \in C \) with Euclidean distance \( d(f(q), f(c)) := \|f(q) - f(c)\|_2 \) and retrieves the top-\( k \) similar images to \( q \) in \( C \). We define a function \( \text{IR}_f(q, C) \), which returns the list of elements in \( C \) ordered by the distance from \( q \). \( \text{IR}_f(q, C) \) denotes the \( j \)-th most similar image to \( q \) in \( C \) and \( \text{IR}_f(q, C)_{\leq j} = \{\text{IR}_f(q, C)_{1}, \ldots, \text{IR}_f(q, C)_{j}\} \) represents the set of images with the first to \( j \)-th highest similarity. We also define a function \( \text{Rank}_f(q, c, C) \) returning the rank of \( c \) in \( \text{IR}_f(q, C) \).

2.2. Adversarial Attacks against CBIR

In recent years, many studies have focused on adversarial attacks on CBIR [38, 40, 12, 22, 26, 39, 14, 28, 1]. These attacks can be categorized into two types of attacks, \textit{query attack (QA)} and \textit{candidate attack (CA)}, depending on whether the AX is given as a query image or a candidate image.

2.2.1 Query Attack (QA)

Let \( C_t \subset C \) be the target candidates in \( C \) specified by the adversary. The adversary aiming at QA perturbs a source query image \( q_s \) to raise or lower the rank of the candidates in \( C_t \). When the attacker’s goal is to raise the rank of the candidates in \( C_t \), adversarial perturbation \( \delta \) for QA is obtained by solving the following optimization problem:

\[
\min_{\delta \in X, \|\delta\|_\infty \leq \epsilon} \sum_{t \in C_t} \text{Rank}_f(q_s + \delta, t, C)
\]
where $\| \cdot \|_\infty$ is an infinite norm and $\epsilon \in \mathbb{R}_{\geq 0}$ is a constant that bounds the size of the perturbation. Eq. (1) cannot be solved directly due to the discrete nature of $\text{Rank}_f(t)$. Instead, [38, 40] minimizes the following objective function:

$$
\min_{\delta \in X, \|\delta\|_\infty \leq \epsilon} \sum_{t \in C_t} \sum_{c \in C} \left[ d(f(q_s + \delta), f(t)) - d(f(q_s + \delta), f(c)) \right] + \epsilon.
$$

Minimization in Eq. (1) is changed to maximization when the attacker’s goal is to lower the rank of the candidates in $C_i$.

### 2.2.2 Candidate Attack (CA)

Let $Q_t = \{q_i \in X\}_{i=1}^M$ be a set of target query images specified by the adversary. The adversary aiming at CA perturbs a source candidate image $c_s \in C$ so that the rank of the perturbed $c_s$ is raised or lowered when $\forall q \in Q_t$ is issued as a query. When the attacker’s goal is to raise the rank of the perturbed $c_s$, adversarial perturbation for CA is obtained by the following minimization problem with respect to $\delta$:

$$
\min_{\delta \in X, \|\delta\|_\infty \leq \epsilon} \sum_{t \in Q_t} \text{Rank}_f(t, c_s + \delta, C).
$$

where $\| \cdot \|_\infty$ is an infinite norm and $\epsilon \in \mathbb{R}_{\geq 0}$ is a constant that bounds the size of the perturbation. Since optimization in Eq. (3) is intractable, [38, 40] optimizes the following objective function instead:

$$
\min_{\delta \in X, \|\delta\|_\infty \leq \epsilon} \sum_{t \in Q_t} \sum_{c \in C} \left[ d(f(t), f(c_s + \delta)) - d(f(t), f(c)) \right].
$$

As well as QA, minimization in Eq. (3) is changed to maximization when the attacker’s goal is to lower the rank of the perturbed $c$.

### 2.3 Certified Robustness

Here, we briefly review the existing definition of the certified robustness and verification algorithm for classification. Then, we define a new certified robustness of CBIR.

#### 2.3.1 Certified Robustness of Classification

The adversarial attacks against the classifier aim to change the predicted label of the classifier to untargeted or targeted label by perturbing the input images [5, 2, 15]. The certified robustness for classification guarantees that predicted labels are kept invariant when the adversarial attacks are limited within a specified range:

**Definition 1** (Certified Robustness of Classification [13]). Let $x \in X$ be an input image and $t \in \{1, \ldots, C\}$ be corresponding label to $x$. Let $f_c : X \rightarrow \mathbb{R}^C$ be a classifier and $f_c(x)_j$ be the logit of class $j \in \{1, \ldots, C\}$ for $x$. Let $\epsilon \in \mathbb{R}_{\geq 0}$. Then, $f_c$ is certified robust at $x$ if $\min_{\delta \in X, \|\delta\|_\infty \leq \epsilon} f_c(x + \delta) - f_c(x)_{i\neq t} > 0$.

### 2.3.2 Verification Algorithm for Classification

Verifying whether $f_c$ satisfies certified robustness of classification at $x$ is reduced to an NP-complete problem [8, 29]. To make the verification computationally tractable, [6, 7, 30, 35, 36, 24, 21, 37, 31] use lower bounds of margins between logits $m_i(x) \leq \min_{\delta \in X, \|\delta\|_\infty \leq \epsilon} f_c(x + \delta)_i - f_c(x + \delta)_{i\neq t}$ instead of the exact margins. $m_i(x)$ can be obtained by computationally tractable algorithms, such as linear relaxations of neural networks [30, 36, 21], utilizing global or local Lipschitz constant of neural networks [24, 37, 31], or interval bound propagation (IBP) [6, 7, 35]. We remark that, samples that actually satisfy Definition 1 can be judged as non-robust because $m_i(x) > 0$ for $\forall i \in \{1, \ldots, C\} \setminus \{t\}$ is a sufficient condition for Definition 1.

### 2.3.3 Certified Robustness of CBIR

Definition 1 is not suitable for CBIR as it is because the goals of adversarial attacks against classification and CBIR are different: the adversarial attacks against classification aim to change the predicted class label of the classifier, whereas QA and CA aim to change the rank of the candidates. Thus, in certified defense for CBIR, we need to consider rank invariance rather than label invariance against AXs. We define the certified robustness of CBIR against QA and CA as follows, respectively:

**Definition 2** $(\alpha, \epsilon)$-Robustness against QA. Let $f : X \rightarrow \mathbb{R}^d$ be a feature extractor. Let $q \in X$ and $C = \{c_i \mid c_i \in X\}_{i=1}^N$ be a query image and a set of candidate images, respectively. Let $\alpha \in \mathbb{N}_0$ and $\epsilon \in \mathbb{R}_{\geq 0}$. Then, for $\forall \delta \in \{\delta \mid \delta \in X, \|\delta\|_\infty \leq \epsilon\}$, $f$ satisfies $(\alpha, \epsilon)$-robust against QA at $\text{IR}_f(q, C)_j$, $q$, and $C$ if

$$
|\text{Rank}_f(q + \delta, \text{IR}_f(q, C)_j, C) - j| \leq \alpha.
$$

**Definition 3** $(\alpha, \epsilon)$-Robustness against CA. Let $f : X \rightarrow \mathbb{R}^d$ be a feature extractor. Let $q \in X$ and $C = \{c_i \mid c_i \in X\}_{i=1}^N$ be a query image and a set of candidate images, respectively. Let $\alpha \in \mathbb{N}_0$ and $\epsilon \in \mathbb{R}_{\geq 0}$. Then, for $\tilde{C} = \{\text{IR}_f(q, C)_i + \delta_i\}_{i=1}^N$ where $\forall \delta_1, \ldots, \forall \delta_N \in \{\delta \mid \delta \in X, \|\delta\|_\infty \leq \epsilon\}$, $f$ satisfies $(\alpha, \epsilon)$-robust against CA at $\text{IR}_f(q, C)_j$, $q$, and $C$ if

$$
|\text{Rank}_f(q, \text{IR}_f(q, C)_j + \delta_j, \tilde{C}) - j| \leq \alpha.
$$

In both robustness definitions, we introduced $\alpha$ to relax the strictness of the guarantee because it can be too stringent to require complete rank invariance.
3. Verification Algorithms for CBIR

In this section, we propose verification algorithms to verify whether given \( f \) satisfies \((\alpha, \epsilon)\)-robustness against QA and CA at given IR\(_f(q, C)_j\), \( q \), and \( C \) (Definition 2 and Definition 3). Since they are computationally intractable, the key challenge of designing the verification algorithms is to make them relax and computationally efficient.

**Overview** Our idea to recover tractability is to introduce computationally tractable sufficient conditions for \((\alpha, \epsilon)\)-robustness against QA and CA. Unfortunately, the existing sufficient condition for Definition 1 cannot be used directly because Definition 2 and Definition 3 depend on distances in the feature space rather than the margins of logits. Thus, in Section 3.1, we first derive sufficient conditions for Definition 2 and Definition 3 using the upper and lower bounds in polynomial time using interval bound propagation (IBP) [6, 7, 35], which is fast and scalable to DNNs.

### 3.1. Sufficient Conditions for \((\alpha, \epsilon)\)-Robustness against QA and CA

In this subsection, we derive sufficient conditions for Definition 2 and Definition 3. Let \( x_1, x_2 \in X \) and \( \epsilon \in \mathbb{R}_{\geq 0} \). Then, we define upper and lower bounds as follows:

\[
\bar{d}_{x_2}(x_1) \geq \max_{\delta \in X} \| f(x_1 + \delta) - f(x_2) \|_2, \quad (7)
\]

\[
\underline{d}_{x_2}(x_1) \leq \min_{\delta \in X} \| f(x_1 + \delta) - f(x_2) \|_2. \quad (8)
\]

We omit \( f \) from the arguments of \( \bar{d}_{x_2}(x_1) \) and \( \underline{d}_{x_2}(x_1) \) for simplicity when it is obvious from the context.

To derive sufficient conditions for Definition 2 and Definition 3, we first derive upper and lower bounds of Rank\(_f(q + \delta, IR_f(q, C)_j, C)\) in Eq. (5) and Rank\(_f(q, IR_f(q, C)_j + \delta_j, C)\) in Eq. (6) by comparing \( \bar{d}_{x_2}(x_1) \) and \( \underline{d}_{x_2}(x_1) \) instead of comparing exact distances:

**Theorem 1** (Upper and Lower Bounds of Rank Under QA).

Let \( f : X \to \mathbb{R}^d \) be a feature extractor. Let \( q \in X \) and \( C = \{c_i | c_i \in X \}_{i=1}^N \) be a query image and a set of candidate images, respectively. Let \( \alpha \in \mathbb{N}_0 \) and \( \epsilon \in \mathbb{R}_{\geq 0} \). Then, for \( \forall \delta \in \{ \delta \mid \delta \in X, \| \delta \|_\infty \leq \epsilon \}, \)

\[
\text{Rank}_f(q + \delta, \hat{C}, C) \leq N - \sum_{c \in C} \mathbb{1}[\bar{d}_c(q) < \underline{d}_c(q)]. \quad (9)
\]

\[
\text{Rank}_f(q + \delta, \hat{C}, C) \geq \sum_{c \in C} \mathbb{1}[\bar{d}_c(q) < \underline{d}_c(q)] + 1 \quad (10)
\]

where \( \hat{c} = IR_f(q, C)_j \).

**Proof.** The proof is shown in Appendix.

### 3.2. Verification Algorithms to Obtain the Upper and Lower Bounds in Polynomial Time

We omit \( f \) from the arguments of \( \bar{d}_{x_2}(x_1) \) and \( \underline{d}_{x_2}(x_1) \) for simplicity when it is obvious from the context.

To derive sufficient conditions for Definition 2 and Definition 3, we first derive upper and lower bounds of Rank\(_f(q + \delta, IR_f(q, C)_j, C)\) in Eq. (5) and Rank\(_f(q, IR_f(q, C)_j + \delta_j, C)\) in Eq. (6) by comparing \( \bar{d}_{x_2}(x_1) \) and \( \underline{d}_{x_2}(x_1) \) instead of comparing exact distances:

**Theorem 2** (Upper and Lower Bounds of Rank Under CA).

Let \( f : X \to \mathbb{R}^d \) be a feature extractor. Let \( q \in X \) and \( C = \{c_i | c_i \in X \}_{i=1}^N \) be a query image and a set of candidate images, respectively. Let \( \alpha \in \mathbb{N}_0 \) and \( \epsilon \in \mathbb{R}_{\geq 0} \). Then, for \( \hat{C} = \{IR_f(q, C)_i + \delta_i \}_{i=1}^N \) where \( \forall \delta_1, ..., \forall \delta_N \in \{ \delta \mid \delta \in X, \| \delta \|_\infty \leq \epsilon \}, \)

\[
\text{Rank}_f(q, \hat{c} + \delta_j, \hat{C}) \leq N - \sum_{c \in C} \mathbb{1}[\bar{d}_c(q) < \underline{d}_c(q)], \quad (11)
\]

\[
\text{Rank}_f(q, \hat{c} + \delta_j, \hat{C}) \geq \sum_{c \in C} \mathbb{1}[\bar{d}_c(q) < \underline{d}_c(q)] + 1 \quad (12)
\]

where \( \hat{c} = IR_f(q, C)_j \).

**Proof.** The proof is shown in Appendix.

### 3.3. Sufficient Conditions for \((\alpha, \epsilon)\)-Robustness against QA

Let \( f : X \to \mathbb{R}^d \) be a feature extractor. Let \( q \in X \) and \( C = \{c_i | c_i \in X \}_{i=1}^N \) be a query image and a set of candidate images, respectively. Let \( \alpha \in \mathbb{N}_0 \) and \( \epsilon \in \mathbb{R}_{\geq 0} \). Then, \( f \) satisfies \((\alpha, \epsilon)\)-robust against QA at IR\(_f(q, C)_j\), \( q \), and \( C \) if

\[
\alpha \geq N - \sum_{c \in C} \mathbb{1}[\bar{d}_c(q) < \underline{d}_c(q)] - j \quad (13)
\]

\[
\land -\alpha \leq \sum_{c \in C} \mathbb{1}[\bar{d}_c(q) < \underline{d}_c(q)] + 1 - j. \quad (14)
\]

**Proof.** The proof is shown in Appendix.

### 3.4. Sufficient Conditions for \((\alpha, \epsilon)\)-Robustness against CA

Let \( f : X \to \mathbb{R}^d \) be a feature extractor. Let \( q \in X \) and \( C = \{c_i | c_i \in X \}_{i=1}^N \) be a query image and a set of candidate images, respectively. Let \( \alpha \in \mathbb{N}_0 \) and \( \epsilon \in \mathbb{R}_{\geq 0} \). Then, \( f \) satisfies \((\alpha, \epsilon)\)-robust against CA at IR\(_f(q, C)_j\), \( q \), and \( C \) if

\[
\alpha \geq N - \sum_{c \in C} \mathbb{1}[\bar{d}_c(q) < \underline{d}_c(q)] - j \quad (13)
\]

\[
\land -\alpha \leq \sum_{c \in C} \mathbb{1}[\bar{d}_c(q) < \underline{d}_c(q)] + 1 - j. \quad (14)
\]

**Proof.** The proof is shown in Appendix.

From Theorem 3 and Theorem 4, we can verify whether Definition 2 and Definition 3 are satisfied in polynomial time if we can calculate \( \bar{d}_{x_2}(x_1) \) and \( \underline{d}_{x_2}(x_1) \) in polynomial time.
3.2. Evaluation of \( \overline{d}_{x_2}(x_1) \) and \( \underline{d}_{x_2}(x_1) \)

Next, we show how to evaluate \( \overline{d}_{x_2}(x_1) \) and \( \underline{d}_{x_2}(x_1) \). Since Eq. (13) and Eq. (14) are sufficient conditions of Definition 2 and Definition 3, respectively, they do not necessarily hold, even when Definition 2 and Definition 3 are guaranteed. Whether Eq. (13) and Eq. (14) can hold depends on the tightness of \( \overline{d}_{x_2}(x_1) \) and \( \underline{d}_{x_2}(x_1) \). For this reason, we need to obtain meaningfully tight evaluation of \( \overline{d}_{x_2}(x_1) \) and \( \underline{d}_{x_2}(x_1) \). In this subsection, we propose methods to calculate \( \overline{d}_{x_2}(x_1) \) and \( \underline{d}_{x_2}(x_1) \) in polynomial time by utilizing Interval Bound Propagation (IBP) \([6, 7, 35]\). IBP is an algorithm for calculating the upper and lower bounds of logsit when a bounded region in the input space is given as input. IBP is used for robustness verification of the classification task and known to give a meaningfully tight bound for this purpose. Since the computational complexity of IBP is comparable to two forward propagations of DNNs, the IBP is scalable even with DNNs.

3.2.1 Original IBP

Given, \( x \in X, \epsilon \in \mathbb{R}_{\geq 0}, \) and \( L \)-layer classifier \( f_c \), original IBP evaluates the upper and lower bounds of \( f_c(x + \delta) \) for \( \forall \delta \in \{ \delta \mid \delta \in X, \| \delta \|_{\infty} \leq \epsilon \} \). Let \( z^l = W_l h^{l-1} + b^l \) be the \( l \)-th affine layer (e.g. fully connected layer and convolution layer) and \( h^{l-1} = \sigma(z^{l-1}) \) be a monotonic activation function (e.g. ReLU) where \( l \in \{1, \ldots, L\} \) and \( h^0 = x \). Then, IBP provides upper and lower bounds on the outputs of \( l \)-th affine layers as follows:

\[
\overline{z}^l = W_l h^{l-1} + b^l \leq W_l \|h^{l-1} - b^l\|_2 + b^l, \quad (15)
\]

\[
\underline{z}^l = W_l h^{l-1} + b^l \geq -W_l \|h^{l-1} - b^l\|_2 + b^l, \quad (16)
\]

where \( | \cdot | \) represents the element-wise absolute value operator, \( h^{l-1} = \sigma(z^{l-1}) \), \( h^0 = x + \epsilon \mathbf{1} \), and \( b^0 = x - \epsilon \mathbf{1} \).

3.2.2 Proposed Methods

We can evaluate \( \overline{d}_{x_2}(x_1) \) and \( \underline{d}_{x_2}(x_1) \) by utilizing IBP. Let \( f(x)_i \) be the \( i \)-th element of \( f(x) \). Let \( \overline{f}(x)_i \) and \( \underline{f}(x)_i \) be upper and lower bounds of \( f(x)_i \) calculated by Eq.(15) and Eq.(16), respectively. Then, we can evaluate \( \overline{d}_{x_2}(x_1) \) and \( \underline{d}_{x_2}(x_1) \) by the following theorems:

**Theorem 5.** Let \( x_1, x_2 \in X \) and \( \epsilon \in \mathbb{R}_{\geq 0} \). Then,

\[
\max_{\delta \in X, \| \delta \|_{\infty} \leq \epsilon} d(f(x_1 + \delta), f(x_2)) \leq \sqrt{\sum_{i \in \{1, \ldots, d\}} \max \{ |\overline{f}(x_1)_i - f(x_2)_i|, |\underline{f}(x_2)_i - \overline{f}(x_1)_i| \}^2 }.
\]

\[ (17) \]

**Proof.** The proof is shown in Appendix. \( \square \)

**Theorem 6.** Let \( x_1, x_2 \in X \) and \( \epsilon \in \mathbb{R}_{\geq 0} \). Then,

\[
\min_{\delta \in X, \| \delta \|_{\infty} \leq \epsilon} d(f(x_1 + \delta), f(x_2)) \geq \sqrt{\sum_{i \in \{1, \ldots, d\}} \min \{0, \overline{f}(x_1)_i - f(x_2)_i, f(x_2)_i - \overline{f}(x_1)_i \}^2 }.
\]

(18)

**Proof.** The proof is shown in Appendix. \( \square \)

The computational complexity of calculating the upper bound in Eq. (17) and the lower bound in Eq. (18) is comparable to three forward propagation of DNNs. Thus, calculating Eq. (17) and Eq. (18) requires one more forward propagation of DNNs than calculating the lower bounds of margins between logit \( m(x) \) by IBP for robustness verification for classification. Evaluating Eq. (17) and Eq. (18) to determine if the derived robustness conditions (Eq. (13) and Eq. (14)) is satisfied, we can obtain our tractable verification algorithms as follows:

\[
\text{Verify}_{\alpha, \epsilon}(f, q, C, j) = \begin{cases} 
\text{True} & \text{if Eq. (13) or (14) is True} \\
\text{False} & \text{otherwise.}
\end{cases}
\]

(19)

4. Robust Training for CBIR

In this section, we propose robustness training for CBIR. We experimentally confirm that Eq.(13) and Eq.(14) are always not satisfied for all \( q, C \), and \( j \) used in our experiments when using \( f \) trained by conventional metric learning (See Section 5 for details). This is because the upper and lower bounds calculated by Eq. (17) and Eq. (18) can be too loose to satisfy the sufficient conditions Eq. (13) and Eq. (14). To increase the number of inputs that satisfy Eq. (13) and Eq. (14), we need to train \( f \) so that attains tighter evaluation of \( \overline{d}_{x_2}(x_1) \) and \( \underline{d}_{x_2}(x_1) \).

To this end, we propose two new objective functions to train feature extractor for CBIR. One is training of general feature extractor that attains tighter bounds in Eq. (17) and Eq. (18) without knowledge of query and candidate images. The other is fine tuning of feature extractor given that candidate images for the target CBIR are provided. We remark that both algorithms are independent, and the latter algorithm can be applied to the feature extractor trained with the former algorithm.

4.1. Training General Feature Extractor for Robust CBIR

Recall that tighter evaluation of the upper bound in Eq. (17) and the lower bound in Eq. (18) is needed to attain certified robustness in a meaningful way. Our idea is to train \( f \)
by simultaneously minimizing conventional objective function (e.g., triplet loss [19]) and the regularization term to make the bounds in Eq. (17) and Eq. (18) tighter.

Let \( D_{\text{train}} = \{(a, p, n)\}_{i=1}^{M} \) be a training data set where \( p \) belongs to the same class as \( a \), and \( n \) belongs to a different class than \( a \). Here, the training dataset and query/candidate images of CBIR are mutually exclusive. Then, our objective function for fine-tuning is given as follows:

\[
\min_{f} \sum_{(a, p, n) \in D_{\text{train}}} \kappa \cdot T(a, p, n) + (1 - \kappa) \cdot \sum_{x \in \{p, n\}} \text{Reg}(a, x) \tag{20}
\]

where \( \text{Reg}(a, x) = \max \left\{ d(f(a), f(x)) - d_{x}(a), \right\} \)

\[
d(f(a), f(x)) - d_{x}(a) \]

and \( T(a, p, n) \) is the triplet loss [19] often used in metric learning, which affects the performance of CBIR. \( \text{Reg}(a, x) \) is a regularization term to encourage that the upper and lower bound of \( \|f(a + \delta) - f(x)\|_2 \) are close to \( \|f(a) - f(x)\|_2 \). \( \kappa \in [0, 1] \) is a hyperparameter to adjust the trade-off between performance of CBIR and \((\alpha, \epsilon)\)-robustness of CBIR against QA and CA. We call the training with Eq. (20) as Tightly Bounding Training (TBT).

### 4.2. Fine-tuning DNNs to Candidate Images

The feature extractor obtained by Eq. (20) is independent of the CBIR query and candidate set. In this subsection, assuming that the candidate images for the target CBIR are given, we show a method to fine tune the feature extractor to the set of candidate images. The objective of this fine-tuning is to reduce the gap between Definition 3 and the corresponding sufficient condition in Eq. (14) by adjusting \( f \) with the given candidate images. To achieve this, we update \( f \) so that tighter evaluation of Eq. (17) and Eq. (18) is attained with given candidate images while maintaining the performance of CBIR.

Let \( C = \{c_i | c_i \in X\}_{i=1}^{N} \) be the set of candidate images. Let \( f_0 \) be the pre-trained feature extractor before fine-tuning. Then, our objective function for fine-tuning is given as follows:

\[
\min_{f} \sum_{c_1, c_2 \in C} \left( \kappa \cdot d(f_0(c_1), f(c_1)) + (1 - \kappa) \cdot \text{Reg}(c_1, c_2) \right) \tag{21}
\]

The first term maintains the accuracy of the CBIR by ensuring that the difference between the features calculated by \( f \) and \( f_0 \) is small. The second term is a regularization term to encourage that the upper and lower bound of \( \|f(c_1 + \delta) - f(c_2)\|_2 \) are close to \( \|f(c_1) - f(c_2)\|_2 \). \( \kappa \in [0, 1] \) is a hyperparameter to adjust the trade-off between the performance of CBIR and \((\alpha, \epsilon)\)-robustness against CA. We call fine-tuning with Eq. (21) as Fine-tuning to Candidates with Tighter Bounds (FCTB).

### 5. Experiments

In this section, we evaluate our proposed robustness training (TBT and FCTB), in terms of CBIR accuracy on clean images and robustness against QA and CA. The robustness is evaluated by empirical robustness, which is the accuracy of CBIR on the generated AXs, and certified robustness, which represents how often CBIR achieves \((\alpha, \epsilon)\)-robustness for given inputs by our robustness verification algorithm Eq. (19).

#### 5.1. Experimental Setting

##### 5.1.1 Datasets

We use MNIST [11], Fashion-MNIST (FMNIST) [32], and CIFAR10 [10] for our evaluations. These datasets consist of an training and test set annotated with labels. We train feature extractors \( f \) on the training sets and evaluate \( f \) using the test set. Let \( Q = \{(q_i, y_{q_i})\}_{i=1}^{Q} \) and \( C = \{(c_i, y_{c_i}) \in X\}_{i=1}^{|C|} \) be the annotated set of query and candidate images, respectively. We randomly select \( Q \) and \( C \) without duplication from the test set so that \( |Q| = 1000 \) and \( |C| = 1000 \). Pixel values of images in all datasets are in \([0, 1]\).

##### 5.1.2 Evaluation Measures

**Performance of CBIR.** To evaluate the performance of CBIR, we use Recall@K, which is one of the evaluation measures for CBIR [18, 27]. Recall@K evaluates whether how often any of the top K candidates is similar to the query image. For evaluation purpose, images belonging to the same class are regarded as similar images. Then, Recall@K is defined as follows:

\[
\frac{1}{|Q|} \sum_{(q_i, y_{q_i}) \in Q} \begin{cases} 
1 & \text{if } \exists(c, y_c) \in \text{IR}_{f}(q_i, C) \leq K \text{ s.t. } y_c = y_{q_i}, \\
0 & \text{otherwise.}
\end{cases} \tag{22}
\]

**Empirical Robustness.** To evaluate the empirical robustness against QA and CA, we extend recall@K and define empirical robust Recall@K (ER-Recall@K) against QA and CA. ER-Recall@K against QA represents how often any of the top K candidates is similar to the query image under QA:

\[
\frac{1}{|Q|} \sum_{(q_i, y_{q_i}) \in Q} \begin{cases} 
1 & \text{if } \exists(c, y_c) \in \text{IR}_{f}(q_i + \delta, C) \leq K \text{ s.t. } y_c = y_{q_i}, \\
0 & \text{otherwise}
\end{cases} \tag{23}
\]

where \( \delta_1, ..., \delta_{|Q|} \) are adversarial perturbations generated with Eq.(2). We randomly select a single target candidate
image \(C_t = \{(c_t, y_{c_t})\} \subseteq C\) such that \(y_{c_t} \neq y_q\), for each \((q_t, y_{q_t}) \in Q\). We minimize Eq.(2) by using PGD [15] with the step size of \(\frac{\epsilon}{10}\) and the number of updates of 100, where \(\epsilon \in \{0.1, 0.2\}\) for MNIST and FMNIST and \(\epsilon \in \{\frac{2}{255}, \frac{4}{255}\}\) for CIFAR10, respectively.

ER-Recall@K against CA represents how often any of the top \(K\) candidates is similar image to the query image under CA:

\[
\frac{1}{|Q|} \sum_{(q, y_q) \in Q} \begin{cases} 
1 & \exists (c, y_c) \in \text{IR}_f(q, C \setminus C_t \cup \tilde{C}_s) \leq K \text{ s.t. } y_c = y_q, \\
0 & \text{otherwise.}
\end{cases}
\]  

(24)

where \(C_s \subseteq C\) is a set of source candidate images, and \(\tilde{C}_s = \{(c_i + \delta_1, y_{c_i}), (c_i, y_{c_i}) \in C_s\}_{i=1}^{100}\) is the set of images obtained by adding adversarial perturbation \(\delta_1, ..., \delta_{|C_s|}\) to each image in \(C_s\) with Eq.(4). We randomly select 100 source candidate images \(C_s = \{(c_i, y_{c_i})\}_{i=1}^{100}\) such that \(y_{c_i} \neq y_q\), for each \((q_t, y_{q_t}) \in Q\). We minimize Eq.(4) using PGD with the same step and perturbation size as the QA.

Certified Robustness. To evaluate the certified robustness, we define an extension of recall@K, certified robust Recall@K (CR-Recall@K). Given a set of query images, this measure evaluates how often (i) the retrieved candidate image by the query image has certified robustness against QA or CA, and (ii) similar to the query image:

\[
\frac{1}{|Q|} \sum_{(q, y_q) \in Q} \begin{cases} 
1 & \exists (c, y_c) \in \text{IR}_f(q, C) \leq K \text{ s.t. } y_c = y_q \land \text{Verify}_{e,\alpha}(f, q, C, \text{Rank}_f(q, c, C)), \\
0 & \text{otherwise.}
\end{cases}
\]  

(25)

where \(\text{Verify}_{e,\alpha}(f, q, C, \text{Rank}_f(q, c, C))\) is defined by Eq. (19). We use \(\alpha = K - \text{Rank}_f(q, c, C)\) for each \(c \in \text{IR}_f(q, C) \leq K\). Then, \(\text{Verify}_{e,\alpha}(f, q, C, \text{Rank}_f(q, c, C))\) verifies whether \(c\) is still included in \(\text{IR}_f(q, C) \leq K\) under QA and CA. and CR-Recall@K is a lower bound of ER-Recall@K. Due to page limitations, only the results for \(e = 0.2\) and \(e = \frac{2}{255}\) are shown here. The results for \(e = 0.1\) and \(e = \frac{3}{255}\) are included in Appendix.

5.1.3 Comparison Methods

We compare our proposed robustness training Eq. (20) (TBT) and Eq. (21) (FCTB) with three existing methods: (i) triplet Loss (Triplet) [19], (ii) anti-collapse triplet (ACT), which is an adversarial training for CBIR to improve empirical robustness [40], (iii) robust training for classification using interval bound propagation (C-IBP) to improve certified robustness of classification task [6].

We use Triplet as a baseline which does not have any mechanism for robustness. We compare TBT and FCTB with ACT to show that adversarial training is not sufficient to improve certified robustness of CBIR. We also compare TBT and FCTB with C-IBP to show that robust training for improving certified robustness for the classification task is inadequate to improve certified robustness for CBIR. Details of each method are explained in Appendix.

5.1.4 Architectures and Training Hyper Parameters

Architectures. In our experiments, we train the feature extractor \(f\) of embedding dimensionality 128 in two different model architectures, as shown in Appendix. We refer to each model as Small (3-layer CNN) and Large (6-layer CNN), respectively. Due to page limitations, the results for Small are included in Appendix.

Hyperparameters. The total number of training epochs is 100 for MNIST and FMNIST and 200 for CIFAR10. We use the Adam optimizer [9] with a batch size of 100 and an initial learning rate of 0.001. We decay the learning rate by times 0.1 at 25 and 42 epochs for MNIST and FMNIST and times 0.5 every 10 epochs between 130 and 200 epochs for CIFAR10. The margin of triplet loss is set to \(m = 1.0\).

When training with TBT, to stabilize training, we use scheduling strategy for \(\epsilon\) and \(\kappa\) proposed in [6]. Specifically, \(\epsilon\) is gradually increased from 0.0 to \(\epsilon_e\), and the \(\kappa\) is gradually decreased from 1.0 to \(\kappa_e\). We use \(\epsilon_e = 0.2\) for MNIST and FMNIST, and \(\epsilon_e = \frac{2}{255}\) for CIFAR10, respectively. We use \(\kappa_e = 0.5\) for all datasets. Then, we linearly increase \(\epsilon\) and decrease \(\kappa\) between \(2K\) and \(10K\) steps. The results of other \(\kappa_e\) are shown in Appendix.

When training with FCTB, we fine-tune the pre-trained feature extractor with TBT. We set fixed \(\epsilon\) to 0.2 for MNIST and FMNIST and \(\epsilon = \frac{2}{255}\) for CIFAR10. We set fixed \(\kappa\) to 0.2 for MNIST and 0.1 for FMNIST and CIFAR10. Other hyperparameters are shown in Appendix.

5.2 Results

Table 1 and Table 2 show the results of Recall@K, and ER-Recall@K, and CR-Recall@K for Large. We can see that TBT has less Recall@K than Triplet, ACT, and C-IBP from Table 1. This is presumably due to the fact that the diversity of feature representation is reduced by making the upper and lower bound evaluated tighter. However, the gap in Recall@K between TBT and the existing methods becomes smaller as \(K\) increases. Thus, that is not a practical problem in situations where \(K\) is large.

From Table 2, we can confirm both ER-Recall@K and CR-Recall@K of Triplet are significantly lower than the other methods. C-IBP and ACT achieve higher ER-Recall@K than Triplet, while their CR-Recall@K is zero or nearly zero, even with larger \(K\). This implies that C-IBP and ACT cannot help to provide certified robustness of CBIR. This is because ACT is training to improve empirical robustness, which is no enough to improve certified robustness. We also conjecture that C-IBP is not sufficient to
Table 1: Comparison of Recall@K (Large). Each value is rounded off to two decimal places.

<table>
<thead>
<tr>
<th>K</th>
<th>MNIST</th>
<th></th>
<th></th>
<th></th>
<th>MNIST</th>
<th></th>
<th></th>
<th></th>
<th>MNIST</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>10</td>
<td>20</td>
<td>40</td>
<td>1</td>
<td>10</td>
<td>20</td>
<td>40</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>Triplet</td>
<td>0.99</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>0.89</td>
<td>0.98</td>
<td>0.99</td>
<td>0.99</td>
<td>0.58</td>
<td>0.93</td>
</tr>
<tr>
<td>ACT</td>
<td>0.99</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>0.83</td>
<td>0.97</td>
<td>0.98</td>
<td>0.99</td>
<td>0.63</td>
<td>0.93</td>
</tr>
<tr>
<td>C-IBP</td>
<td>0.97</td>
<td>0.99</td>
<td>1.00</td>
<td>1.00</td>
<td>0.75</td>
<td>0.96</td>
<td>0.98</td>
<td>0.99</td>
<td>0.39</td>
<td>0.87</td>
</tr>
<tr>
<td>TBT</td>
<td>0.94</td>
<td>0.98</td>
<td>0.99</td>
<td>0.99</td>
<td>0.62</td>
<td>0.93</td>
<td>0.97</td>
<td>0.98</td>
<td>0.18</td>
<td>0.81</td>
</tr>
<tr>
<td>TBT+FCTB</td>
<td>0.93</td>
<td>0.98</td>
<td>0.98</td>
<td>0.99</td>
<td>0.64</td>
<td>0.94</td>
<td>0.97</td>
<td>0.98</td>
<td>0.19</td>
<td>0.82</td>
</tr>
</tbody>
</table>

Table 2: Comparison of empirical robust (ER) Recall@K and certified robust (CR) Recall@K (Large). QA and CA represents query attack and candidate attack, respectively. For calculating ER-Recall@K and CR-Recall@K, we use $\epsilon = 0.2$ (MNIST and FMNIST) and $\epsilon = \frac{2}{255}$ (CIFAR10). Each value is rounded off to two decimal places.

<table>
<thead>
<tr>
<th>K</th>
<th>ER-Recall@K (QA)</th>
<th>ER-Recall@K (CA)</th>
<th>CR-Recall@K (QA)</th>
<th>CR-Recall@K (CA)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>10</td>
<td>20</td>
<td>40</td>
</tr>
<tr>
<td>Triplet</td>
<td>0.00</td>
<td>0.05</td>
<td>0.09</td>
<td>0.14</td>
</tr>
<tr>
<td>ACT</td>
<td>0.97</td>
<td>0.99</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>C-IBP</td>
<td>0.97</td>
<td>0.99</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>TBT</td>
<td>0.92</td>
<td>0.98</td>
<td>0.98</td>
<td>0.99</td>
</tr>
<tr>
<td>TBT+FCTB</td>
<td>0.92</td>
<td>0.97</td>
<td>0.98</td>
<td>0.99</td>
</tr>
<tr>
<td>MNIST</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ER-Recall@K (QA)</td>
<td>0.00</td>
<td>0.09</td>
<td>0.14</td>
<td>0.20</td>
</tr>
<tr>
<td>CR-Recall@K (QA)</td>
<td>0.74</td>
<td>0.95</td>
<td>0.97</td>
<td>0.98</td>
</tr>
<tr>
<td>CR-Recall@K (CA)</td>
<td>0.71</td>
<td>0.96</td>
<td>0.98</td>
<td>0.99</td>
</tr>
<tr>
<td>TBT</td>
<td>0.59</td>
<td>0.93</td>
<td>0.97</td>
<td>0.98</td>
</tr>
<tr>
<td>TBT+FCTB</td>
<td>0.60</td>
<td>0.93</td>
<td>0.97</td>
<td>0.99</td>
</tr>
<tr>
<td>FMNIST</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ER-Recall@K (QA)</td>
<td>0.19</td>
<td>0.70</td>
<td>0.82</td>
<td>0.89</td>
</tr>
<tr>
<td>CR-Recall@K (QA)</td>
<td>0.47</td>
<td>0.88</td>
<td>0.94</td>
<td>0.97</td>
</tr>
<tr>
<td>CR-Recall@K (CA)</td>
<td>0.40</td>
<td>0.87</td>
<td>0.94</td>
<td>0.98</td>
</tr>
<tr>
<td>C-IBP</td>
<td>0.20</td>
<td>0.79</td>
<td>0.92</td>
<td>0.97</td>
</tr>
<tr>
<td>TBT</td>
<td>0.19</td>
<td>0.81</td>
<td>0.93</td>
<td>0.97</td>
</tr>
<tr>
<td>TBT+FCTB</td>
<td>0.19</td>
<td>0.81</td>
<td>0.93</td>
<td>0.97</td>
</tr>
</tbody>
</table>

6. Limitations

A drawback of our certified defense is that it does not scale to high-resolution images, which require advanced architecture. We train feature extractor with TBT using CUB-200-2011 [25] (image size is $224 \times 224$) and VGG architecture [20]. The detail of experimental settings is explained in Appendix. As a result, its training collapses, which means that the trained feature extractor returns the same value for all test inputs. This is because IBP provides very loose bounds for advanced deep architectures, resulting in extremely large regularization terms in Eq.(20). We also obtain the same results when training a feature extractor with C-IBP. Developing a certified defense for CBIR that scales to high-resolution images is a future research direction.

7. Conclusion

In this study, we proposed a certified defense for CBIR. Our certified defense improves the certified robustness of CBIR, which guarantees that no AX that largely changes the ranking of CBIR exists around the query or candidate images. To the best of our knowledge, this is the first paper on certified defense for CBIR.

Acknowledgment

This work is partly supported by Japan science and technology agency (JST), CREST JPMJCR21D3, and Japan society for the promotion of science (JSPS), Grants-in-Aid for Scientific Research 22H00521 and 19H04164.
References


[29] Lily Weng, Huan Zhang, Hongge Chen, Zhao Song, Chouei Hsieh, Luca Daniel, Duane Boning, and Inderjit Dhillon.


