

Adversarial Ranking Attack and Defense

Mo Zhou¹, Zhenxing Niu², Le Wang³, Qilin Zhang⁴, Gang Hua⁵

¹Xidian University, ²Alibaba DAMO MIIL, ³Xi'an Jiaotong University, ⁴HERE Technologies, ⁵Wormpex AI Research
 {cdluminate, zhenxingniu, samqzhang, ganghua}@gmail.com, lewang@xjtu.edu.cn

Abstract

Deep Neural Network (DNN) classifiers are vulnerable to adversarial attack, where an imperceptible perturbation could result in misclassification. However, the vulnerability of DNN-based image ranking systems remains under-explored. In this paper, we propose two attacks against deep ranking systems, i.e., Candidate Attack and Query Attack, that can raise or lower the rank of chosen candidates by adversarial perturbations. Specifically, the expected ranking order is first represented as a set of inequalities, and then a triplet-like objective function is designed to obtain the optimal perturbation. Conversely, a defense method is also proposed to improve the ranking system robustness, which can mitigate all the proposed attacks simultaneously. Our adversarial ranking attacks and defense are evaluated on datasets including MNIST, Fashion-MNIST, and Stanford-Online-Products. Experimental results demonstrate that a typical deep ranking system can be effectively compromised by our attacks. Meanwhile, the system robustness can be moderately improved with our defense. Furthermore, the transferable and universal properties of our adversary illustrate the possibility of realistic black-box attack.

1. Introduction

Despite the successful application in computer vision tasks such as image classification [35, 25], Deep Neural Networks (DNNs) have been found vulnerable to adversarial attacks. In particular, the DNN’s prediction can be arbitrarily changed by just applying an imperceptible perturbation to the input image [75, 21]. Moreover, such adversarial attacks can effectively compromise the recent state-of-the-art DNNs such as Inception [73, 74] and ResNet [25]. This poses a serious security risk on many DNN-based applications such as face recognition, where recognition evasion or impersonation can be easily achieved [15, 70, 34, 78].

Previous adversarial attacks primarily focus on *classification*, however, we speculate that DNN-based image ranking systems [5, 8, 76, 33, 56, 19, 39] also suffer from similar vulnerability. Taking the image-based product search as

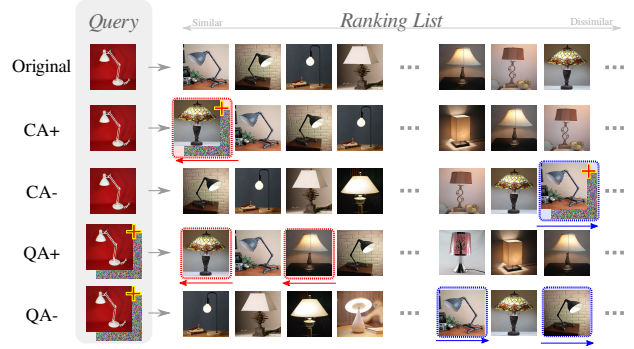


Figure 1. Adversarial ranking attack that can raise or lower the rank of chosen candidates by adversarial perturbations. In Candidate Attack, adversarial perturbation is added to the candidate and its rank is raised (CA+) or lowered (CA-). In Query Attack, adversarial perturbation is added to the query image, and the ranks of chosen candidates are raised (QA+) or lowered (QA-).

an example, a fair ranking system should rank the database products according to their visual similarity to the query, as shown in Fig. 1 (row 1). Nevertheless, a malicious seller may attempt to raise the rank of his/her own product by adding perturbation to the image (CA+, row 2), or lower the rank of his competitor’s product (CA-, row 3); Besides, a “man-in-the-middle” attacker (e.g., a malicious advertising company) could hijack and imperceptibly perturb the query image in order to promote (QA+, row 4) or impede (QA-, row 5) the sales of specific products.

Unlike classification tasks where images are predicted independently, the rank of one candidate is related to the query as well as other candidates for image ranking. The relative relations among candidates and queries determine the final ranking order. Therefore, we argue that the existing adversarial classification attacks are incompatible with the ranking scenario. Thus, we need to thoroughly study the *adversarial ranking attack*.

In this paper, adversarial ranking attack aims to *raise* or *lower* the ranks of some chosen candidates $C = \{c_1, c_2, \dots, c_m\}$ with respect to a specific query set $Q = \{q_1, q_2, \dots, q_w\}$. This can be achieved by either Candidate Attack (CA) or Query Attack (QA). In particular, CA is de-

defined as to raise (*abbr.* CA+) or lower (*abbr.* CA-) the rank of a single candidate c with respect to the query set Q by perturbing c itself; while QA is defined as to raise (*abbr.* QA+) or lower (*abbr.* QA-) the ranks of a candidate set C with respect to a single query q by perturbing q . Thus, adversarial ranking attack can be achieved by performing CA on each $c \in C$, or QA on each $q \in Q$. In practice, the choice of CA or QA depends on the accessibility to the candidate or query respectively, *i.e.*, CA is feasible for modifiable candidate, while QA is feasible for modifiable query.

An effective implementation of these attacks is proposed in this paper. As we know, a typical DNN-based ranking model maps objects (*i.e.*, queries and candidates) to a common embedding space, where the distances among them determine the final ranking order. Predictably, the object’s position in the embedding space will be changed by adding a perturbation to it. Therefore, the essential of adversarial ranking attack is to find a proper perturbation, which could push the object to a desired position that leads to the expected ranking order. Specifically, we first represent the expected ranking order as a set of inequalities. Subsequently, a triplet-like objective function is designed according to those inequalities, and combined with Projected Gradient Descent (PGD) to efficiently obtain the desired adversarial perturbation.

Opposed to the proposed attacks, *adversarial ranking defense* is worth being investigated especially for security-sensitive deep ranking applications. Until now, the Madry defense [50] is regarded as the most effective method for classification defense. However, we empirically discovered a primary challenge of diverging training loss while directly adapting such mechanism for ranking defense, possibly due to the generated adversarial examples being too “strong”. In addition, such defense mechanism needs to defend against distinct ranking attacks individually, but a *generic* defense method against all CA+, CA-, QA+ and QA- attacks is preferred.

To this end, a shift-distance based ranking defense is proposed, which could simultaneously defend against all attacks. Note that the position shift of objects in the embedding space is the key for all ranking attacks. Although different attacks prefer distinct shift directions (*e.g.*, CA+ and CA- often prefer opposed shifting directions), a large shift distance is their common preference. If we could reduce the shift distance of embeddings incurred by adversarial perturbation, all attacks can be simultaneously defended. Specifically, we first propose a shift-distance based ranking attack, which aims to push the objects as far from their original positions as possible. And then, the adversarial examples generated from such attack is involved in the adversarial training. Experimental results manifest that our ranking defense can converge and moderately improve model robustness.

In addition, our ranking attacks have some good proper-

ties for realistic applications. First, our adversary is transferable, *i.e.*, the adversary obtained from a known DNN ranker can be directly used to attack an unknown DNN ranker (*i.e.*, the network architecture and parameters are unknown). Second, our attacks can be extended to *universal* ranking attacks with slight performance drop, *i.e.*, we could learn a *universal* perturbation to all candidates for CA, or a *universal* perturbation to all queries for QA. Such properties illustrate the possibility of practical black-box attack.

To the best of our knowledge, this is the first work that thoroughly studies the adversarial ranking attack and defense. In brief, our contributions are:

1. The adversarial ranking attack is defined and implemented, which can intentionally change the ranking results by perturbing the candidates or queries.
2. An adversarial ranking defense method is proposed to improve the ranking model robustness, and mitigate all the proposed attacks simultaneously.

2. Related Works

Adversarial Attacks. Szegedy *et al.* [75] claimed that DNN is susceptible to imperceptible adversarial perturbations added to inputs, due to the intriguing “blind spot” property, which was later ascribed to the local linearity [21] of neural networks. Following these findings, many white-box (model architecture and parameters are known to the adversary) attacking methods [54, 61, 36, 7, 10, 13, 66, 72, 50, 80, 9, 20] are proposed to effectively compromise the state-of-the-art DNN classifiers. Among them, PGD [50] is regarded as one of the most powerful attacks [2]. Notably, adversarial examples are discovered to be transferable [60, 59] among different neural network classifiers, which inspired a series of black-box attacks [71, 79, 83, 45, 14, 28]. On the other hand, universal (*i.e.*, image-agnostic) adversarial perturbations are also discovered [53, 41]. The existence of adversarial examples stimulated research interests in areas such as object detection [48, 11, 85], semantic segmentation [1], and automatic speech recognition [65], *etc.* It is even possible to create physical adversarial examples [70, 4, 18, 78].

Deep Ranking. Different from the traditional “learning to rank” [42, 31] methods, DNN-based ranking methods often embed data samples (including both queries and candidates) of all modalities into a common embedding space, and subsequently determine the ranking order based on distance. Such workflow has been adopted in distance metric learning [8, 76, 57, 30], image retrieval [5], cross-modal retrieval [56, 19, 39, 33], and face recognition [67].

Adversarial Attacks in Deep Ranking. For information retrieval and ranking systems, the risk of malicious users manipulating the ranking always exists [23, 27]. However, only a few research efforts have been made in ad-

versarial attacks in deep ranking. Liu *et al.* [46] proposed adversarial queries leading to incorrect retrieval results; while Li *et al.* [40] staged similar attack with universal perturbation that corrupts listwise ranking results. None of the aforementioned research efforts explore the *adversarial ranking attack*. Besides, adaptation of distance-based attacks (e.g. [66]) are unsuitable for our scenario.

Adversarial Defenses. Adversarial attacks and defenses are consistently engaged in an arms race [84]. Gradient masking-based defenses can be circumvented [3]. Defensive distillation [58, 62] has been compromised by C&W [7, 6]. As claimed in [26], ensemble of weak defenses are insufficient against adversarial examples. Notably, as an early defense method [75], adversarial training [21, 50, 29, 16, 37, 69, 77, 86, 55] remains to be one of the most effective defenses. Other types of defenses include adversarial detection [47, 52], input transformation/reconstruction/replacement [64, 49, 24, 51, 17], randomization [44, 43], feature denoising [82], network verification [32, 22], evidential deep learning [68], *etc.* However, defense in deep ranking systems remains mostly uncharted.

3. Adversarial Ranking

Generally, a DNN-based ranking task could be formulated as a metric learning problem. Given the query q and candidate set $X = \{c_1, c_2, \dots, c_n\}$, deep ranking is to learn a mapping f (usually implemented as a DNN) which maps all candidates and query into a common embedding space, such that the relative distances among the embedding vectors could satisfy the expected ranking order. For instance, if candidate c_i is more similar to the query q than candidate c_j , it is encouraged for the mapping f to satisfy the inequality $\|f(q) - f(c_i)\| < \|f(q) - f(c_j)\|$ ¹, where $\|\cdot\|$ denotes ℓ_2 norm. For brevity, we denote $\|f(q) - f(c_i)\|$ as $d(q, c_i)$ in following text.

Therefore, adversarial ranking attack is to find a proper adversarial perturbation which leads the ranking order to be changed as expected. For example, if a less relevant c_j is expected to be ranked *ahead* of a relevant c_i , it is desired to find a proper perturbation r to perturb c_j , *i.e.* $\tilde{c}_j = c_j + r$, such that the inequality $d(q, c_i) < d(q, c_j)$ could be changed into $d(q, c_i) > d(q, \tilde{c}_j)$. In the next, we will describe Candidate Attack and Query Attack in detail.

3.1. Candidate Attack

Candidate Attack (CA) aims to raise (*abbr.* CA+) or lower (*abbr.* CA-) the rank of a *single* candidate c with respect to a set of queries $Q = \{q_1, q_2, \dots, q_w\}$ by adding perturbation r to the candidate itself, *i.e.* $\tilde{c} = c + r$.

Let $\text{Rank}_X(q, c)$ denote the rank of the candidate c with respect to the query q , where X indicates the set of all candi-

dates, and a smaller rank value represents a higher ranking. Thus, the CA+ that *raises* the rank of c with respect to every query $q \in Q$ by perturbation r could be formulated as the following problem,

$$r = \arg \min_{r \in \Gamma} \sum_{q \in Q} \text{Rank}_X(q, c + r), \quad (1)$$

$$\Gamma = \{r \mid \|r\|_\infty \leq \varepsilon; r, c + r \in [0, 1]^N\}, \quad (2)$$

where Γ is a ℓ_∞ -bounded ε -neighbor of c , $\varepsilon \in [0, 1]$ is a predefined small positive constant, the constraint $\|r\|_\infty \leq \varepsilon$ limits the perturbation r to be “visually imperceptible”, and $c + r \in [0, 1]^N$ ensures the adversarial example remains a valid input image. Although alternative “imperceptible” constraints exist (e.g., ℓ_0 [72, 12], ℓ_1 [10] and ℓ_2 [7, 54] variants), we simply follow [21, 36, 50] and use the ℓ_∞ constraint.

However, the optimization problem Eq. (1)–(2) cannot be directly solved due to the discrete nature of the rank value $\text{Rank}_X(q, c)$. In order to solve the problem, a surrogate objective function is needed.

In metric learning, given two candidates $c_p, c_n \in X$ where c_p is ranked ahead of c_n , *i.e.* $\text{Rank}_X(q, c_p) < \text{Rank}_X(q, c_n)$, the ranking order is represented as an inequality $d(q, c_p) < d(q, c_n)$ and formulated in triplet loss:

$$L_{\text{triplet}}(q, c_p, c_n) = [\beta + d(q, c_p) - d(q, c_n)]_+, \quad (3)$$

where $[\cdot]_+$ denotes $\max(0, \cdot)$, and β is a manually defined constant margin. This function is known as the triplet (*i.e.* pairwise ranking) loss [8, 67].

Similarly, the attacking goal of CA+ in Eq. (1) can be readily converted into a series of inequalities, and subsequently turned into a sum of triplet losses,

$$L_{\text{CA+}}(c, Q; X) = \sum_{q \in Q} \sum_{x \in X} [d(q, c) - d(q, x)]_+. \quad (4)$$

In this way, the original problem in Eq. (1)–(2) can be reformulated into the following constrained optimization problem:

$$r = \arg \min_{r \in \Gamma} L_{\text{CA+}}(c + r, Q; X). \quad (5)$$

To solve the optimization problem, Projected Gradient Descent (PGD) method [50, 36] (*a.k.a* the iterative version of FGSM [21]) can be used. Note that PGD is one of the most effective first-order gradient-based algorithms [2], popular among related works about adversarial attack.

Specifically, in order to find an adversarial perturbation r to create a desired adversarial candidate $\tilde{c} = c + r$, the PGD algorithm alternates two steps at every iteration $t = 1, 2, \dots, \eta$. Step one updates \tilde{c} according to the gradient of Eq. (4); while step two clips the result of step one to fit in the ε -neighboring region Γ :

$$\tilde{c}_{t+1} = \text{Clip}_{c, \Gamma} \{\tilde{c}_t - \alpha \text{sign}(\nabla_{\tilde{c}_t} L_{\text{CA+}}(\tilde{c}_t, Q, X))\}, \quad (6)$$

¹Sometimes cosine distance is used instead.

where α is a constant hyper-parameter indicating the PGD step size, and \tilde{c}_1 is initialized as c . After η iterations, the desired adversarial candidate \tilde{c} is obtained as \tilde{c}_η , which is optimized to satisfy as many inequalities as possible. Each inequality represents a pairwise ranking sub-problem, hence the adversarial candidate \tilde{c} will be ranked ahead of other candidates with respect to every specified query $q \in Q$.

Likewise, the **CA-** that *lowers* the rank of a candidate c with respect to a set of queries Q can be obtained in similar way:

$$L_{\text{CA-}}(c, Q; X) = \sum_{q \in Q} \sum_{x \in X} [-d(q, c) + d(q, x)]_+. \quad (7)$$

3.2. Query Attack

Query Attack (**QA**) is supposed to raise (*abbr.* **QA+**) or lower (*abbr.* **QA-**) the rank of a set of candidates $C = \{c_1, c_2, \dots, c_m\}$ with respect to the query q , by adding adversarial perturbation r to the query $\tilde{q} = q + r$. Thus, **QA** and **CA** are two “symmetric” attacks. The **QA-** for *lowering* the rank could be formulated as follows:

$$r = \arg \max_{r \in \Gamma} \sum_{c \in C} \text{Rank}_X(q + r, c), \quad (8)$$

where Γ is the ε -neighbor of q . Likewise, this attacking objective can also be transformed into the following constrained optimization problem:

$$L_{\text{QA-}}(q, C; X) = \sum_{c \in C} \sum_{x \in X} [-d(q, c) + d(q, x)]_+, \quad (9)$$

$$r = \arg \min_{r \in \Gamma} L_{\text{QA-}}(q + r, C; X), \quad (10)$$

and it can be solved with the PGD algorithm. Similarly, the **QA+** loss function $L_{\text{QA+}}$ for *raising* the rank of c is as follows:

$$L_{\text{QA+}}(q, C; X) = \sum_{c \in C} \sum_{x \in X} [d(q, c) - d(q, x)]_+. \quad (11)$$

Unlike **CA**, **QA** perturbs the *query* image, and hence may drastically change its semantics, resulting in abnormal retrieval results. For instance, after perturbing a “lamp” query image, some unrelated candidates (*e.g.*, “shelf”, “toaster”, *etc.*) may appear in the top return list. Thus, an ideal query attack should preserve the query semantics, *i.e.*, the candidates in $X \setminus C$ ² should retain their original ranks if possible. Thus, we propose the Semantics-Preserving Query Attack (**SP-QA**) by adding the **SP** term to mitigate the semantic changes q , *e.g.*,

$$L_{\text{SP-QA-}}(q, C; X) = L_{\text{QA-}}(q, C; X) + \xi L_{\text{QA+}}(q, C_{\text{SP}}; X), \quad (12)$$

where $C_{\text{SP}} = \{c \in X \setminus C | \text{Rank}_{X \setminus C}(q, c) \leq G\}$, *i.e.*, C_{SP} contains the top- G most-relevant candidates corresponding

²The complement of the set C .

to q , and the $L_{\text{QA+}}(q, C_{\text{SP}}; X)$ term helps preserve the query semantics by retaining some C_{SP} candidates in the retrieved ranking list. Constant G is a predefined integer; and constant ξ is a hyper-parameter for balancing the attack effect and semantics preservation. Unless mentioned, in the following text **QA** means **SP-QA** by default.

3.3. Robustness & Defense

Adversarial defense for classification has been extensively explored, and many of them follows the adversarial training mechanism [29, 37, 50]. In particular, the adversarial counterparts of the original training samples are used to replace or augment the training samples. Until now, Madry defense [50] is regarded as the most effective [77, 3] adversarial training method. However, when directly adapting such classification defense to improve ranking robustness, we empirically discovered a primary challenge of diverging training loss, possibly due to the generated adversarial examples being too “strong”. Moreover, such defense mechanism needs to defend against distinct attacks individually. Therefore, a *generic* defense against all the proposed attacks is preferred.

Note that the underlying principle of adversarial ranking attack is to shift the embeddings of candidates/queries to a proper place, and a successful attack depends on a large shift distance as well as a correct shift direction. A large shift distance is an indispensable objective for all the **CA+**, **CA-**, **QA+** and **QA-** attacks. Predictably, a reduction in shift distance could improve model robustness against all attacks simultaneously.

To this end, we propose a “maximum-shift-distance” attack that pushes an embedding vector as far from its original position as possible (resembles Feature Adversary [66] for classification), $r = \arg \max_{r \in \Gamma} d(c + r, c)$. Then we use adversarial examples obtained from this attack to replace original training samples for adversarial training, hence reduce the shift distance incurred by adversarial perturbations.

A ranking model can be normally trained with the defensive version of the triplet loss:

$$L_{\text{d-t}}(q, c_p, c_n) = L_{\text{triplet}} \left(q + \arg \max_{r \in \Gamma} d(q + r, q), \right. \\ \left. c_p + \arg \max_{r \in \Gamma} d(c_p + r, c_p), \right. \\ \left. c_n + \arg \max_{r \in \Gamma} d(c_n + r, c_n) \right). \quad (13)$$

Unlike the direct adaptation of Madry defense, the training loss does converge in our experiments.

4. Experiments

To validate the proposed attacks and defense, we use three commonly used ranking datasets including MNIST [38], Fashion-MNIST [81], and Stanford Online

ε	CA+				CA-				QA+				QA-			
	$w=1$	2	5	10	$w=1$	2	5	10	$m=1$	2	5	10	$m=1$	2	5	10
(CTD) Cosine Distance, Triplet Loss (R@1=99.1%)																
0	50	50	50	50	2.1	2.1	2.1	2.1	50	50	50	50	0.5	0.5	0.5	0.5
0.01	44.6	45.4	47.4	47.9	3.4	3.2	3.1	3.1	45.2	46.3	47.7	48.5	0.9	0.7	0.6	0.6
0.03	33.4	37.3	41.9	43.9	6.3	5.9	5.7	5.6	35.6	39.2	43.4	45.8	1.9	1.4	1.1	1.1
0.1	12.7	17.4	24.4	30.0	15.4	14.9	14.8	14.7	14.4	21.0	30.6	37.2	5.6	4.4	3.7	3.5
0.3	2.1	9.1	13.0	17.9	93.9	93.2	93.0	92.9	6.3	11.2	22.5	32.1	8.6	6.6	5.3	4.8

Table 1. Adversarial ranking attack on vanilla model with MNIST. The “+” attacks (*i.e.* CA+ and QA+) raise the rank of chosen candidates towards 0 (%); while the “-” attacks (*i.e.* CA- and QA-) lower the ranks of chosen candidates towards 100 (%). Applying $\varepsilon = 0.01, 0.03, 0.1, 0.3$ QA+ attacks on the model, the SP term keeps the ranks of C_{SP} no larger than 3.6%, 5.7%, 7.7%, 7.7%, respectively, regardless of m . With the QA- counterpart, the ranks of C_{SP} are kept no larger than 1.6%, 1.6%, 1.5%, 1.5%, respectively, regardless of m . For all the numbers in the table, “%” is omitted.

Product (SOP) [57]. We respectively train models on these datasets with PyTorch [63], and conduct attacks³ on their corresponding test datasets (used as X).

Evaluation Metric. Adversarial ranking attack aims to change the ranks of candidates. For each candidate c , its *normalized* rank is calculated as $R(q, c) = \frac{\text{Rank}_X(q, c)}{|X|} \times 100\%$ where $c \in X$, and $|X|$ is the length of full ranking list. Thus, $R(q, c) \in [0, 1]$, and a top ranked c will have a small $R(q, c)$. The attack effectiveness can be measured by the magnitude of change in $R(q, c)$.

Performance of Attack. To measure the performance of a single CA attack, we average the rank of candidate c across every query $q \in Q$, *i.e.*, $R_{CA}(c) = \sum_{q \in Q} R(q, c)/w$. Similarly, the performance of a single QA attack can be measured by the average rank across every candidate $c \in C$, *i.e.*, $R_{QA}(q) = \sum_{c \in C} R(q, c)/m$. For the overall performance of an attack, we conduct T times of independent attacks and report the mean of $R_{CA}(c)$ or $R_{QA}(q)$, accordingly.

CA+ & QA+. For CA+, the query set Q is randomly sampled from X . Likewise, for QA+, the candidate set C is from X . Without attack, both the $R_{CA}(c)$ and $R_{QA}(q)$ will approximate to 50%, and the attacks should significantly *decrease* the value.

CA- & QA-. In practice, the Q for CA- and the C for QA- cannot be randomly sampled, because the two attacks are often to lower some top ranked candidates. Thus, the two sets should be selected from the top ranked samples (top-1% in our experiments) in X . Formally, given the candidate c for CA-, we randomly sample the w queries from $\{q \in X | R(c, q) \leq 1\%\}$ as Q . Given the query q for QA-, m candidates are randomly sampled from $\{c \in X | R(q, c) \leq 1\%\}$ as C . Without attack, both the $R_{CA}(c)$ and $R_{QA}(q)$ will be close to 0%, and the attacks should significantly *increase* the value.

Hyper-Parameters. We conduct CA with $w \in$

³Specifically, we use PGD without random starts [50].

ε	CA+				CA-				QA+				QA-			
	$w=1$	2	5	10	$w=1$	2	5	10	$m=1$	2	5	10	$m=1$	2	5	10
(CTD) Cosine Distance, Triplet Loss, Defensive (R@1=98.3%)																
0	50	50	50	50	2.0	2.0	2.0	2.0	50	50	50	50	0.5	0.5	0.5	0.5
0.01	48.9	49.3	49.4	49.5	2.2	2.2	2.2	2.1	49.9	49.5	49.5	49.7	0.5	0.5	0.5	0.5
0.03	47.4	48.4	48.6	48.9	2.5	2.5	2.4	2.4	48.0	48.5	49.2	49.5	0.6	0.6	0.5	0.5
0.1	42.4	44.2	45.9	46.7	3.8	3.6	3.5	3.4	43.2	45.0	47.4	48.2	1.0	0.8	0.7	0.7
0.3	30.7	34.5	38.7	40.7	7.0	6.7	6.5	6.5	33.2	37.2	42.3	45.1	2.4	1.9	1.6	1.5

Table 2. Adversarial ranking defense with MNIST. Applying $\varepsilon = 0.01, 0.03, 0.1, 0.3$ QA+ attacks on model, the ranks of candidates in C_{SP} are kept no larger than 0.5%, 0.5%, 0.5%, 0.5%, respectively, regardless of m . With the QA- counterpart, the ranks of C_{SP} are kept less than 0.4%, 0.4%, 0.4%, 0.4%, respectively, regardless of m .

$\{1, 2, 5, 10\}$ queries, and QA with $m \in \{1, 2, 5, 10\}$ candidates, respectively. In QA, we let $G = 5$. The SP balancing parameter ξ is set to 1 for QA+, and 10^2 for QA-. In addition, We investigate attacks of different strength ε , *i.e.* 0.01, 0.03, 0.1, 0.3 on MNIST and Fashion-MNIST following [50], and 0.01, 0.03, 0.06 on SOP following [37]. The PGD step size is empirically set to $\alpha = \min(\max(\frac{\varepsilon}{10}, \frac{1}{255}), 0.01)$, and the number of PGD iterations to $\eta = \min(\max(10, \frac{2\varepsilon}{\alpha}), 30)$. We perform $T = |X|$ times of attack to obtain the reported performance.

Adversarial Defense. Ranking models are trained using Eq. (13) with the strongest adversary following the procedure of Madry defense [50].

4.1. MNIST Dataset

Following conventional settings with the MNIST [38] dataset, we train a CNN ranking model comprising 2 convolutional layers and 1 fully-connected layer. This CNN architecture (denoted as C2F1) is identical to the one used in [50] except for the removal of the last fully-connected layer. Specifically, the ranking model is trained with cosine distance and triplet loss. The retrieval performance of the model is Recall@1=99.1% (R@1), as shown in Tab. 1 in grey highlight.

Attacking results against this vanilla model (*i.e.*, the ranking model which is not enhanced with our defense method) are presented in Tab. 1. For example, a strong CA+ attack (*i.e.*, $\varepsilon = 0.3$) for $w = 1$ can raise the rank $R_{CA}(c)$ from 50% to 2.1%. Likewise, the rank of C can be raised to 9.1%, 13.0%, 17.9% for $w = 2, 5, 10$ chosen queries, respectively.

On the other hand, a strong CA- attack for $w = 1$ can lower the rank $R_{CA}(c)$ from 2.1% to 93.9%. The results of strong CA- attacks for $w = 2, 5, 10$ are similar to the $w = 1$ case.

The results of QA+ and QA- are also shown in Tab. 1. the rank changes with QA attacks are less dramatic (but still significant) than CA. This is due to the additional difficulty introduced by SP term in Eq. (12), and the QA attack effectiveness is inversely correlated with ξ . For instance, a strong QA- for $m = 1$ can only lower the rank $R_{QA}(q)$ from 0.5%

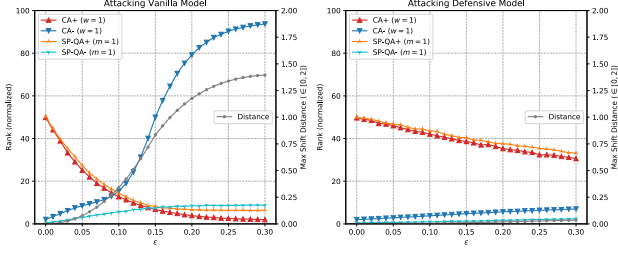


Figure 2. Comparison of Attacks on vanilla and defensive models. Apart from the ranks of chosen candidates, We also measure the maximum shift distance of embedding vectors that adversarial perturbation could incur.

to 8.6%, but the attacking effect can be further boosted by decreasing ξ . More experimental results are presented in following discussion. In brief, our proposed attacks against the vanilla ranking model is effective.

Next, we evaluate the performance of our defense method. Our defense should be able to enhance the robustness of a ranking model, which can be measured by the difference between the attack effectiveness with our defense and the attack effectiveness without our defense. As a common phenomenon of adversarial training, our defense mechanism leads to a slight retrieval performance degradation for unperturbed input (highlighted in blue in Tab. 2), but the attacking effectiveness is clearly mitigated by our defense. For instance, the same strong **CA+** attack for $w = 1$ on the defensive model (*i.e.*, the ranking model which is enhanced by our defense method) can only raise the rank $R_{CA}(c)$ from 50% to 30.7%, compared to its vanilla counterpart raising to 2.1%. Further analysis suggests that the weights in the first convolution layer of the defensive model are closer to 0 and have smaller variance than those of the vanilla model, which may help resist adversarial perturbation from changing the layer outputs into the local linear area of ReLU [21].

To visualize the effect of our attacks and defense, we track the attacking effect with ϵ varying from 0.0 to 0.3 on the vanilla and defensive models, as shown in Fig. 2. It is noted that our defense could significantly suppress the maximum embedding shift distance incurred by adversarial perturbation to nearly 0, but the defensive model is still not completely immune to attacks. We speculate the defensive model still has “blind spots” [75] in some local areas that could be exploited by the attacks.

In summary, these results and further experiments suggest that: (1) deep ranking models are vulnerable to adversarial ranking attacks, no matter what loss function or distance metric is selected; (2) vanilla models trained with contrastive loss are more robust than those trained with triplet loss. This is possibly due to contrastive loss explicitly reducing the intra-class embedding variation. Addition-

ϵ	CA+				CA-				QA+				QA-			
	$w=1$	2	5	10	$w=1$	2	5	10	$m=1$	2	5	10	$m=1$	2	5	10
(CT) Cosine Distance, Triplet Loss (R@1=88.8%)																
0	50	50	50	50	1.9	1.9	1.9	1.9	50	50	50	50	0.5	0.5	0.5	0.5
0.01	36.6	39.9	43.2	44.8	5.6	5.1	4.9	4.8	39.4	42.0	45.3	47.1	2.1	1.6	1.2	1.1
0.03	19.7	25.4	31.7	35.6	15.5	14.8	14.4	14.3	21.7	28.2	35.7	40.6	5.6	4.1	3.3	2.9
0.1	3.7	10.5	17.3	22.7	87.2	86.7	86.3	86.3	7.1	12.4	23.6	32.5	10.9	8.3	6.7	6.0
0.3	1.3	9.4	16.0	21.5	100.0	100.0	100.0	100.0	6.3	10.8	21.8	31.7	12.6	9.4	7.5	6.6
(CTD) Cosine Distance, Triplet Loss, Defensive (R@1=79.6%)																
0	50	50	50	50	1.2	1.2	1.2	1.2	50	50	50	50	0.5	0.5	0.5	0.5
0.01	48.9	48.9	49.3	49.3	1.4	1.4	1.4	1.4	49.4	49.9	49.9	50.0	0.5	0.5	0.5	0.5
0.03	47.1	47.9	48.3	48.3	2.0	1.9	1.8	1.8	48.3	49.1	49.5	49.8	0.7	0.6	0.6	0.6
0.1	42.4	43.5	44.5	44.8	4.6	4.2	4.0	3.9	45.4	47.2	48.7	49.2	1.4	1.2	1.1	1.1
0.3	32.5	35.4	37.5	38.2	11.2	10.5	10.1	10.0	39.3	42.6	46.5	47.8	3.9	3.3	3.0	2.9

Table 3. Adversarial ranking attack and defense on Fashion-MNIST. The lowest ranks of C_{SP} are 3.0%, 5.2%, 7.8%, 8.3% in QA+, and 1.9%, 1.9%, 1.9%, 1.8% for QA+, respectively.

ally, our defense method could consistently improve the robustness of all these models; (3) different distance metrics have almost negligible contribution on robustness. Specifically, Euclidean distance-based models are slightly more susceptible to weak (*e.g.*, $\epsilon = 0.03$) attacks; (4) Euclidean distance-based models are harder to defend than cosine distance-based ones. Beyond these experiments, we also find that the margin hyper-parameter β of triplet loss and the dimensionality of the embedding space have marginal influences on model robustness.

4.2. Fashion-MNIST Dataset

Fashion-MNIST [81] is an MNIST-like but more difficult dataset, comprising 60,000 training examples and 10,000 test samples. The samples are 28×28 greyscale images covering 10 different fashion product classes, including “T-shirt” and “dress”, *etc.* We train the vanilla and defensive models based on the cosine distance and triplet loss and conduct attack experiments.

The attack and defense results are available in Tab. 3. From the table, we note that our attacks could achieve better effect compared to experiments on MNIST. For example, in a strong **CA+** for $w = 1$, the rank $R_{CA}(c)$ can be raised to 1.3%. On the other hand, despite the moderate improvement in robustness, the defensive model performs worse in unperturbed sample retrieval, as expected. The performance degradation is more pronounced on this dataset compared to MNIST. We speculate the differences are related to the increased dataset difficulty.

4.3. Stanford Online Products Dataset

Stanford Online Products (SOP) dataset [57] contains 120k images of 23k classes of real online products from eBay for metric learning. We use the same dataset split as used in the original work [57]. We also train the same vanilla ranking model using the same triplet ranking loss function with Euclidean distance, except that the GoogLeNet [73] is replaced with ResNet-18 [25]. The ResNet-18 achieves better retrieval performance.

Attack and defense results on SOP are present in Tab. 4.

ε	CA+				CA-				QA+				QA-			
	$w=1$	2	5	10	$w=1$	2	5	10	$m=1$	2	5	10	$m=1$	2	5	10
(ET) Euclidean Distance, Triplet Loss (R@1=63.1%)																
0	50	50	50	50	1.9	1.9	1.9	1.9	50	50	50	50	0.5	0.5	0.5	0.5
0.01	0.0	0.8	2.0	2.6	99.7	99.6	99.4	99.3	4.8	7.0	16.3	25.8	54.9	40.2	27.1	21.9
0.03	0.0	0.3	1.0	1.5	100.0	100.0	100.0	100.0	1.6	3.3	10.0	19.2	68.1	52.4	36.6	30.1
0.06	0.0	0.2	1.0	1.5	100.0	100.0	100.0	100.0	1.1	2.7	8.8	17.6	73.8	57.9	40.3	32.4
(ETD) Euclidean Distance, Triplet Loss, Defensive (R@1=46.4%)																
0	50	50	50	50	2.0	2.0	2.0	2.0	50	50	50	50	0.5	0.5	0.5	0.5
0.01	7.5	12.2	16.5	18.0	66.4	62.6	59.3	57.8	16.1	24.8	36.1	41.4	26.7	18.1	12.2	10.2
0.03	0.7	4.5	8.7	10.4	91.7	90.2	89.1	88.4	7.9	14.5	27.2	35.6	43.4	31.7	21.9	18.1
0.06	0.1	3.8	7.9	9.7	97.3	96.8	96.4	96.2	6.9	12.5	24.3	33.4	51.4	39.0	28.0	23.5

Table 4. Adversarial ranking attack and defense on SOP. With different ε , the worst ranks of C_{SP} in QA+ are 0.2%, 0.7%, 2.0%, 3.3%, and those for QA- are 0.4%, 0.7%, 0.8%, 1.0%, respectively.

It is noted that our attacks are quite effective on this difficult large-scale dataset, as merely 1% perturbation ($\varepsilon = 0.01$) to any candidate image could make it ranked ahead or behind of nearly all the rest candidates (as shown by the CA+ and CA- results with $w = 1$). The QA on this dataset is significantly effective as well. On the other hand, our defense method leads to decreased retrieval performance, *i.e.* R@1 from 63.1% to 46.4%, which is expected on such a difficult dataset. Meanwhile, our defense could moderately improve the model robustness against relatively weaker adversarial examples (*e.g.* $\varepsilon = 0.01$), but improving model robustness on this dataset is more difficult, compared to experiments on other datasets.

By comparing the results among all the three datasets, we find ranking models trained on harder datasets more susceptible to adversarial attack, and more difficult to defend. Therefore, we speculate that models used in realistic applications could be easier to attack, because they are usually trained on larger-scale and more difficult datasets.

5. Discussions

In this section, we study the transferability of our adversarial ranking examples, and universal adversarial perturbation for ranking. Both of them illustrate the possibility of practical black-box attack. Additionally, we also perform parameter search on the balancing parameter ξ for QA.

5.1. Adversarial Example Transferability

As demonstrated in previous experiments, deep ranking models can be compromised by our white-box attacks. In realistic scenarios, the white-box attacks are not practical enough because the model to be attacked is often unknown (*i.e.*, the architecture and parameters are unknown).

On the other hand, adversarial examples for classification have been found transferable [60, 59] (*i.e.* model-agnostic) between different models with different network architectures. And the transferability has become the foundation of a class of existing black-box attacks. Specifically, for such a typical attack, adversarial examples are generated from a replacement model [60] using a white-box attack,

CA+ Transfer (Black Box), $w = 1$					QA+ Transfer (Black Box), $m = 1$				
From	To				From	To			
		LeNet	C2F1	Res18			LeNet	C2F1	Res18
LeNet		50→16.6	35.1	34.3	LeNet		50→20.5	43.0	45.8
C2F1		28.6	50→2.1	31.3	C2F1		43.5	50→6.3	45.4
Res18		24.4	27.0	50→2.2	Res18		41.4	40.4	50→14.1

CA- Transfer (Black Box), $w = 1$					QA- Transfer (Black Box), $m = 1$				
From	To				From	To			
		LeNet	C2F1	Res18			LeNet	C2F1	Res18
LeNet		2.5→63.7	2.1→10.0	2.1→9.1	LeNet		0.5→7.0	0.5→1.6	0.5→1.8
C2F1		2.5→9.1	2.1→93.9	2.1→9.3	C2F1		0.5→1.0	0.5→8.6	0.5→1.9
Res18		2.5→9.9	2.1→11.8	2.1→66.7	Res18		0.5→0.8	0.5→1.2	0.5→6.9

Table 5. Transferability of adversarial ranking examples. Adversarial examples are generated from one model and directly used on another. We report the rank of the same c with respect to the same q across different models to illustrate the transfer attack effectiveness. Transferring adversarial examples to a model itself (the diagonal lines) is equivalent to white-box attack.

and are directly used to attack the black-box model.

Adversarial ranking attack could be more practical if the adversarial ranking examples have the similar transferability. Besides the C2F1 model, we train two vanilla models on the MNIST dataset: (1) LeNet [38], which has lower model capacity compared to C2F1; (2) ResNet-18 [25] (denoted as Res18), which has a better network architecture and higher model capacity.

The results are present in Tab. 5. For example, in the CA+ transfer attack, we generate adversarial candidates from the C2F1 model and directly use them to attack the Res18 model (row 2, column 3, top-left table), and the ranks of the adversarial candidates with respect to the same query is still raised to 31.3%. We also find the CA- transfer attack is effective, where the ranks of our adversarial candidates are lowered, *e.g.* from 2.1% to 9.3% (row 2, column 3, bottom-left table). Similar results can be observed on the QA transfer experiments, and they show weaker effect due to the SP term.

From the results, we find that: (1) CNN with better architecture and higher model capacity (*i.e.*, Res18) is less susceptible to adversarial ranking attack. This conclusion is consistent with Madry’s [50], which claims that higher model capacity could help improve model robustness; (2) adversarial examples generated from the Res18 have the most significant effectiveness in transfer attack; (3) CNN of low model capacity (*i.e.*, LeNet), performs moderately in terms of both adversarial example transferability and model robustness. We speculate its robustness stems from a forced regularization effect due low model capacity. Beyond these, adversarial ranking examples are transferable disregarding the difference in loss function or distance metric.

Apart from transferability across different architectures, we also investigated the transferability between the C2F1 models with different network parameters. Results suggest similar transferability between these models. Notably, when transferring adversarial examples to a defensive C2F1 model, the attacking effect is significantly mitigated. The

CA+	CA-	QA+	QA-
50 → 2.1	2.1 → 93.9	50 → 0.2	0.5 → 94.1
I-CA+	I-CA-	I-QA+	I-QA-
50 → 18.1	0.6 → 9.5	50 → 20.5	2.1 → 7.6
I-CA+ (unseen)	I-CA- (unseen)	I-QA+ (unseen)	I-QA- (unseen)
50 → 18.5	0.7 → 9.4	50 → 21.0	2.2 → 7.4

Table 6. Universal Adversarial Perturbation for Ranking on MNIST. Each pair of results presents the original rank of chosen candidates and that after adding adversarial perturbation. Both w , m are set to 1. Parameter ξ is set to 0 to reduce attack difficulty.

result further demonstrates the effectiveness of our defense.

5.2. Universal Perturbation for Ranking

Recently, universal (*i.e.* image-agnostic) adversarial perturbation [53] for classification has been found possible, where a single perturbation may lead to misclassification when added to any image. Thus, we also investigate the existence of universal adversarial perturbation for ranking.

To this end, we follow [53] and formulate the image-agnostic CA+ (*abbr.* **I-CA+**). Given a set of candidates $C = \{c_1, c_2, \dots, c_m\}$ and a set of queries $Q = \{q_1, q_2, \dots, q_w\}$, **I-CA+** is to find a *single* universal adversarial perturbation r , so that the rank of every perturbed candidate $\tilde{c} = c + r$ ($c \in C$) with respect to Q can be raised. The corresponding optimization problem of **I-CA+** is:

$$r = \arg \min_{r \in \Gamma} \sum_{c \in C} L_{CA+}(c + r, Q; X). \quad (14)$$

When applied with such universal perturbation, the rank of any candidate *w.r.t* Q is expected to be raised. The objective functions of **I-CA-**, **I-QA+** and **I-QA-** can be obtained in similar way. Note, unlike [40] which aims to find universal perturbation that can make image retrieval system return irrelevant results, our universal perturbations have distinct purposes.

We conduct experiment on the MNIST dataset. For **I-CA+** attack, we randomly sample 5% of X for generating the universal perturbation. Following [53], another non-overlapping 5% examples are randomly sampled from X to test whether the generated perturbation is generalizable on “unseen” (*i.e.*, not used for generating the perturbation) images. Experiments for the other image-agnostic attacks are conducted similarly. Note, we only report the **I-CA-** and **I-QA-** effectiveness on the 1% top ranked samples, similar to **CA-** and **QA-**.

As shown in Tab. 13, our **I-CA** can raise the ranks of C to 18.1%, or lower them to 9.5%. When added to “unseen” candidate images, our universal perturbation could retain nearly the same effectiveness. This may due to low intra-class variance of the MNIST dataset.

ξ	QA+				QA-			
	m=1	2	5	10	m=1	2	5	10
(CT) Cosine distance, Triplet loss								
0	0.2, 33.6	6.3, 23.7	18.5, 26.5	29.6, 25.7	94.1, 89.4	93.2, 90.3	92.6, 90.9	92.3, 91.2
10^0	6.3, 3.6	11.2, 5.7	22.5, 7.7	32.1, 7.7	55.5, 35.6	52.4, 37.6	50.2, 39.3	49.4, 40.0
10^2	14.1, 0.6	20.8, 0.7	31.2, 0.7	38.1, 0.7	8.6, 1.6	6.6, 1.6	5.3, 1.5	4.8, 1.5
10^4	37.9, 0.1	42.6, 0.1	46.3, 0.1	47.8, 0.1	1.9, 0.1	1.4, 0.1	1.2, 0.1	1.1, 0.1

Table 7. Parameter search on Semantics-Preserving balancing parameter ξ with MNIST. We report two mean ranks in each cell: one for the chosen candidates C ; another for C_{SP} used for SP.

5.3. Semantics Preserving for QA

As discussed previously, the Query Attack (**QA**) may drastically change the semantics of the query q . To alleviate this problem, the Semantics-Preserving (SP) term is added to the naive **QA** to help preserve the query semantics. Predictably, it is more difficult to perform **QA** with a large ξ , as the ranks of C_{SP} are almost not allowed to be changed.

To investigate the actual influence of the balancing parameter ξ , we provide parameter search on it with MNIST dataset. In particular, We set ξ to 0, 10^0 , 10^2 , 10^4 , and compare their results. Note that when $\xi = 0$, the QA becomes *naive* QA as the SP term is eliminated. With a strong SP constant, *e.g.* $\xi = 10^4$, the semantics of the chosen query is almost not allowed to be changed, hence result in extreme difficulty of attack.

As shown in Tab. 7, setting ξ to 0 could greatly boost the attacking effect, but consequently the ranks of C_{SP} will be drastically changed. In contrast, when ξ is set to the excessive value 10^4 for a perfectly stealth QA, the attack can still raise the rank of chosen candidate from 50% to 37.9% in QA+ with $m = 1$, or lower the rank of chosen candidate from 0.5% to 1.9% in QA- with $m = 1$. During these attacks, the ranks of C_{SP} are kept within 0.1 despite of the extreme difficulty. It means the query semantics can be preserved. In practice, we empirically set the parameter ξ as 1 for QA+, or as 10^2 for QA- for the balance between attack effectiveness and preserving query semantics.

6. Conclusion

Deep ranking models are vulnerable to adversarial perturbations that could intentionally change the ranking result. In this paper, we define and implement *adversarial ranking attack* that can compromise deep ranking models. We also propose an *adversarial ranking defense* that can significantly suppress embedding shift distance and moderately improve the ranking model robustness. Moreover, the transferability of our adversarial examples and the existence of universal adversarial perturbations for ranking attack illustrate the possibility of practical black-box attack and potential risk of realistic ranking applications. In the potential of future work, we may explore better ranking loss functions, defenses, and black-box attacks.

Acknowledgements

This work was supported partly by National Key R&D Program of China Grant 2018AAA0101400, NSFC Grants 61629301, 61773312, 61976171, and 61672402, China Postdoctoral Science Foundation Grant 2019M653642, Young Elite Scientists Sponsorship Program by CAST Grant 2018QNR0001, and Natural Science Foundation of Shaanxi Grant 2020JQ-069.

References

- [1] Anurag Arnab, Ondrej Miksik, and Philip HS Torr. On the robustness of semantic segmentation models to adversarial attacks. In *CVPR*, pages 888–897, 2018. 2
- [2] Anish Athalye and Nicholas Carlini. On the robustness of the cvpr 2018 white-box adversarial example defenses. *arXiv preprint arXiv:1804.03286*, 2018. 2, 3
- [3] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018. 3, 4
- [4] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. *arXiv preprint arXiv:1707.07397*, 2017. 2
- [5] Tu Bui, L Ribeiro, Moacir Ponti, and John Collomosse. Compact descriptors for sketch-based image retrieval using a triplet loss convolutional neural network. *CVIU*, 164:27–37, 2017. 1, 2
- [6] Nicholas Carlini and David Wagner. Defensive distillation is not robust to adversarial examples. *arXiv preprint arXiv:1607.04311*, 2016. 3
- [7] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57. IEEE, 2017. 2, 3
- [8] Gal Chechik, Varun Sharma, Uri Shalit, and Samy Bengio. Large scale online learning of image similarity through ranking. *JMLR*, 11(Mar):1109–1135, 2010. 1, 2, 3
- [9] Jianbo Chen and Michael I Jordan. Boundary attack++: Query-efficient decision-based adversarial attack. *arXiv preprint arXiv:1904.02144*, 2019. 2
- [10] Pin-Yu Chen, Yash Sharma, Huan Zhang, Jinfeng Yi, and Cho-Jui Hsieh. Ead: elastic-net attacks to deep neural networks via adversarial examples. In *AAAI*, 2018. 2, 3
- [11] Shang-Tse Chen, Cory Cornelius, Jason Martin, and Duen Horng Polo Chau. Shapeshifter: Robust physical adversarial attack on faster r-cnn object detector. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 52–68. Springer, 2018. 2
- [12] Francesco Croce and Matthias Hein. Sparse and imperceivable adversarial attacks. In *ICCV*, pages 4724–4732, 2019. 3
- [13] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *CVPR*, June 2018. 2
- [14] Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu. Evading defenses to transferable adversarial examples by translation-invariant attacks. In *CVPR*, pages 4312–4321, 2019. 2
- [15] Yinpeng Dong, Hang Su, Baoyuan Wu, Zhifeng Li, Wei Liu, Tong Zhang, and Jun Zhu. Efficient decision-based black-box adversarial attacks on face recognition. In *CVPR*, pages 7714–7722, 2019. 1
- [16] Yinpeng Dong, Hang Su, Jun Zhu, and Fan Bao. Towards interpretable deep neural networks by leveraging adversarial examples. *arXiv preprint arXiv:1708.05493*, 2017. 3
- [17] Abhimanyu Dubey, Laurens van der Maaten, Zeki Yalniz, Yixuan Li, and Dhruv Mahajan. Defense against adversarial images using web-scale nearest-neighbor search. In *CVPR*, pages 8767–8776, 2019. 3
- [18] Kevin Eykholt, Ivan Evtimov, Earlene Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning models. *arXiv preprint arXiv:1707.08945*, 2017. 2
- [19] Fartash Faghri, David J Fleet, Jamie Ryan Kiros, and Sanja Fidler. Vse++: Improved visual-semantic embeddings. *arXiv preprint arXiv:1707.05612*, 2(7):8, 2017. 1, 2
- [20] Aditya Ganeshan and R Venkatesh Babu. Fda: Feature disruptive attack. In *ICCV*, pages 8069–8079, 2019. 2
- [21] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 1, 2, 3, 6, 26, 28
- [22] Divya Gopinath, Guy Katz, Corina S Pasareanu, and Clark Barrett. DeepSAFE: A data-driven approach for checking adversarial robustness in neural networks. *arXiv preprint arXiv:1710.00486*, 2017. 3
- [23] Gregory Goren, Oren Kurland, Moshe Tennenholtz, and Fiana Raiber. Ranking robustness under adversarial document manipulations. In *ACM SIGIR*, pages 395–404. ACM, 2018. 2
- [24] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens Van Der Maaten. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*, 2017. 3
- [25] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, June 2016. 1, 6, 7
- [26] Warren He, James Wei, Xinyun Chen, Nicholas Carlini, and Dawn Song. Adversarial example defense: Ensembles of weak defenses are not strong. In *11th USENIX Workshop on Offensive Technologies (WOOT 17)*, 2017. 3
- [27] Xiangnan He, Zhankui He, Xiaoyu Du, and Tat-Seng Chua. Adversarial personalized ranking for recommendation. In *ACM SIGIR*, pages 355–364. ACM, 2018. 2
- [28] Qian Huang, Zeqi Gu, Isay Katsman, Horace He, Pian Pawakapan, Zhiqiu Lin, Serge Belongie, and Ser-Nam Lim. Intermediate level adversarial attack for enhanced transferability. *arXiv preprint arXiv:1811.08458*, 2018. 2
- [29] Ruitong Huang, Bing Xu, Dale Schuurmans, and Csaba Szepesvári. Learning with a strong adversary. *CoRR*, abs/1511.03034, 2015. 3, 4
- [30] Pierre Jacob, David Picard, Aymeric Histace, and Edouard Klein. Metric learning with horde: High-order regularizer for deep embeddings. In *ICCV*, pages 6539–6548, 2019. 2
- [31] Thorsten Joachims. Optimizing search engines using click-through data. In *ACM SIGKDD*, pages 133–142. ACM, 2002. 2

- [32] 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. 3
- [33] Ryan Kiros, Ruslan Salakhutdinov, and Richard S Zemel. Unifying visual-semantic embeddings with multimodal neural language models. *arXiv preprint arXiv:1411.2539*, 2014. 1, 2
- [34] Stepan Komkov and Aleksandr Petiushko. Advhat: Real-world adversarial attack on arcface face id system. *arXiv preprint arXiv:1908.08705*, 2019. 1
- [35] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, pages 1097–1105, 2012. 1
- [36] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016. 2, 3
- [37] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016. 3, 4, 5
- [38] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 4, 5, 7
- [39] Kuang-Huei Lee, Xi Chen, Gang Hua, Houdong Hu, and Xiaodong He. Stacked cross attention for image-text matching. In *ECCV*, pages 201–216, 2018. 1, 2
- [40] Jie Li, Rongrong Ji, Hong Liu, Xiaopeng Hong, Yue Gao, and Qi Tian. Universal perturbation attack against image retrieval. In *ICCV*, pages 4899–4908, 2019. 3, 8
- [41] Hong Liu, Rongrong Ji, Jie Li, Baoshang Zhang, Yue Gao, Yongjian Wu, and Feiyue Huang. Universal adversarial perturbation via prior driven uncertainty approximation. In *ICCV*, pages 2941–2949, 2019. 2
- [42] Tie-Yan Liu et al. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval*, 3(3):225–331, 2009. 2
- [43] Xuanqing Liu, Minhao Cheng, Huan Zhang, and Cho-Jui Hsieh. Towards robust neural networks via random self-ensemble. In *ECCV*, pages 369–385, 2018. 3
- [44] Xuanqing Liu, Yao Li, Chongruo Wu, and Cho-Jui Hsieh. Adv-bnn: Improved adversarial defense through robust bayesian neural network. *arXiv preprint arXiv:1810.01279*, 2018. 3
- [45] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770*, 2016. 2
- [46] Zhuoran Liu, Zhengyu Zhao, and Martha Larson. Who’s afraid of adversarial queries?: The impact of image modifications on content-based image retrieval. In *ICMR*, pages 306–314. ACM, 2019. 2
- [47] Jiajun Lu, Theerasit Issaranon, and David Forsyth. Safetynet: Detecting and rejecting adversarial examples robustly. In *ICCV*, pages 446–454, 2017. 3
- [48] Jiajun Lu, Hussein Sibai, Evan Fabry, and David Forsyth. No need to worry about adversarial examples in object detection in autonomous vehicles. *arXiv preprint arXiv:1707.03501*, 2017. 2
- [49] Yan Luo, Xavier Boix, Gemma Roig, Tomaso Poggio, and Qi Zhao. Foveation-based mechanisms alleviate adversarial examples. *arXiv preprint arXiv:1511.06292*, 2015. 3
- [50] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017. 2, 3, 4, 5, 7, 28
- [51] Dongyu Meng and Hao Chen. Magnet: a two-pronged defense against adversarial examples. In *ACM SIGSAC*, pages 135–147. ACM, 2017. 3
- [52] Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations. *arXiv preprint arXiv:1702.04267*, 2017. 3
- [53] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *CVPR*, pages 1765–1773, 2017. 2, 7, 8
- [54] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *CVPR*, pages 2574–2582, 2016. 2, 3
- [55] Chaithanya Kumar Mummadi, Thomas Brox, and Jan Hendrik Metzen. Defending against universal perturbations with shared adversarial training. In *ICCV*, pages 4928–4937, 2019. 3
- [56] Zhenxing Niu, Mo Zhou, Le Wang, Xinbo Gao, and Gang Hua. Hierarchical multimodal lstm for dense visual-semantic embedding. In *ICCV*, pages 1881–1889, 2017. 1, 2
- [57] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *CVPR*, pages 4004–4012, 2016. 2, 4, 6
- [58] Nicolas Papernot and Patrick McDaniel. On the effectiveness of defensive distillation. *arXiv preprint arXiv:1607.05113*, 2016. 3
- [59] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016. 2, 7
- [60] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pages 506–519. ACM, 2017. 2, 7
- [61] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 372–387. IEEE, 2016. 2
- [62] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 582–597. IEEE, 2016. 3
- [63] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. *None*, 2017. 4

- [64] Aaditya Prakash, Nick Moran, Solomon Garber, Antonella DiLillo, and James Storer. Deflecting adversarial attacks with pixel deflection. In *CVPR*, pages 8571–8580, 2018. 3
- [65] Yao Qin, Nicholas Carlini, Ian Goodfellow, Garrison Cottrell, and Colin Raffel. Imperceptible, robust, and targeted adversarial examples for automatic speech recognition. *arXiv preprint arXiv:1903.10346*, 2019. 2
- [66] Sara Sabour, Yanshuai Cao, Fartash Faghri, and David J Fleet. Adversarial manipulation of deep representations. *arXiv preprint arXiv:1511.05122*, 2015. 2, 3, 4, 27
- [67] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, pages 815–823, 2015. 2, 3
- [68] Murat Sensoy, Lance Kaplan, and Melih Kandemir. Evidential deep learning to quantify classification uncertainty. In *NeurIPS*, pages 3179–3189, 2018. 3
- [69] Uri Shaham, Yutaro Yamada, and Sahand Negahban. Understanding adversarial training: Increasing local stability of supervised models through robust optimization. *Neurocomputing*, 307:195–204, 2018. 3
- [70] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *ACM SIGSAC*, pages 1528–1540. ACM, 2016. 1, 2
- [71] Yucheng Shi, Siyu Wang, and Yahong Han. Curls & whey: Boosting black-box adversarial attacks. *arXiv preprint arXiv:1904.01160*, 2019. 2
- [72] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 2019. 2, 3
- [73] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, pages 1–9, 2015. 1, 6
- [74] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, pages 2818–2826, 2016. 1
- [75] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013. 1, 2, 3, 6
- [76] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu. Learning fine-grained image similarity with deep ranking. In *CVPR*, pages 1386–1393, 2014. 1, 2
- [77] Jianyu Wang and Haichao Zhang. Bilateral adversarial training: Towards fast training of more robust models against adversarial attacks. In *ICCV*, pages 6629–6638, 2019. 3, 4
- [78] Zhibo Wang, Siyan Zheng, Mengkai Song, Qian Wang, Alireza Rahimpour, and Hairong Qi. advpattern: Physical-world attacks on deep person re-identification via adversarially transformable patterns. In *ICCV*, pages 8341–8350, 2019. 1, 2
- [79] Lei Wu, Zhanxing Zhu, Cheng Tai, et al. Understanding and enhancing the transferability of adversarial examples. *arXiv preprint arXiv:1802.09707*, 2018. 2
- [80] Chaowei Xiao, Jun-Yan Zhu, Bo Li, Warren He, Mingyan Liu, and Dawn Song. Spatially transformed adversarial examples. *arXiv preprint arXiv:1801.02612*, 2018. 2
- [81] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017. 4, 6
- [82] Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan L Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. In *CVPR*, pages 501–509, 2019. 3
- [83] Cihang Xie, Zhishuai Zhang, Yuyin Zhou, Song Bai, Jianyu Wang, Zhou Ren, and Alan L Yuille. Improving transferability of adversarial examples with input diversity. In *CVPR*, pages 2730–2739, 2019. 2
- [84] Xiaoyong Yuan, Pan He, Qile Zhu, and Xiaolin Li. Adversarial examples: Attacks and defenses for deep learning. *IEEE TNNLS*, 2019. 3
- [85] Haichao Zhang and Jianyu Wang. Towards adversarially robust object detection. In *ICCV*, pages 421–430, 2019. 2
- [86] Yaoyao Zhong and Weihong Deng. Adversarial learning with margin-based triplet embedding regularization. In *ICCV*, pages 6549–6558, 2019. 3

A. Adversarial Example Visualization

Some adversarial ranking examples are presented in this section. Every figure contains three rows of pictures. **The first row** shows c , r , $\tilde{c} = c + r$ for CA, or q , r , $\tilde{q} = q + r$ for QA. **The second row** shows the query and the original retrieval results, as well as the chosen candidate c and its immediately adjacent candidates. **The third row** shows the effects of the attack on the ranking list, i.e., either the chosen candidate c is replaced with \tilde{c} for CA, or the query q is replaced with \tilde{q} for QA. **The digits** above every picture is the value of $\text{Rank}_X(q, \cdot) - 1$ ⁴.

Pictures with a “★” mark on the top-left corner are adversarial examples. The “←” indicates the chosen candidate whose rank will be **raised**. The “→” indicates the chosen candidate whose rank will be **lowered**.

Full visualization results including those on SOP (removed from the Arxiv version due to file size limit) can be found at <https://github.com/cdluminate/advrank-pub>.

A.1. MNIST Dataset

CA+. See Fig. 3,4,5. **CA-**. See Fig. 6,7,8.

QA+. See Fig. 9,10,11. **QA-**. See Fig. 12,13,14.

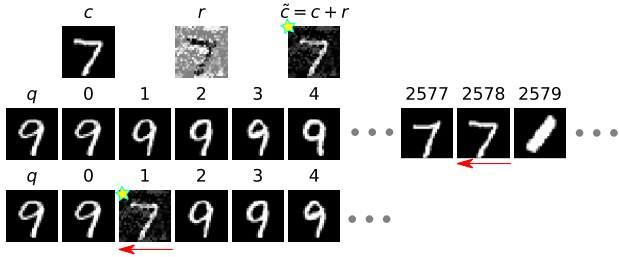


Figure 3. CA+ on MNIST. Example 1. For query “9”, the candidate “7” is ranked at the 2576-th position in the original ranking list (row 2). After adversarial ranking attack, the rank of the perturbed candidate “7” is raised to the 1-st position (row 3). The original candidate “7”, the learned perturbation r , and the perturbed candidate “7” are illustrated in the row 1.

A.2. Fashion-MNIST

CA+. See Fig. 15,16,17. **CA-**. See Fig. 18,19,20.

QA+. See Fig. 21,22,23. **QA-**. See Fig. 24,25,26.

A.3. Semantic-Preserving for QA

Conducting Query Attack without preserving the query semantics will often lead to irrelevant retrieval results at the top of the ranking list, as shown in Fig. 27, which raises red flags and possibly reveals the attack. Therefore, the value

⁴In mathematical context the rank counts from 1, but in implementation it counts from 0 instead. Hence the offset.

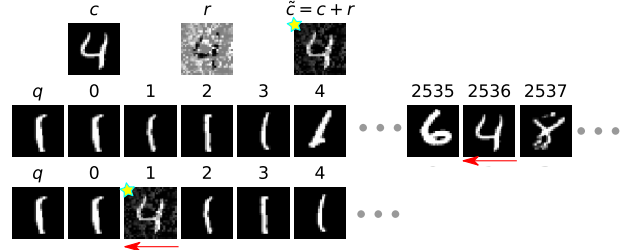


Figure 4. CA+ on MNIST. Example 2.

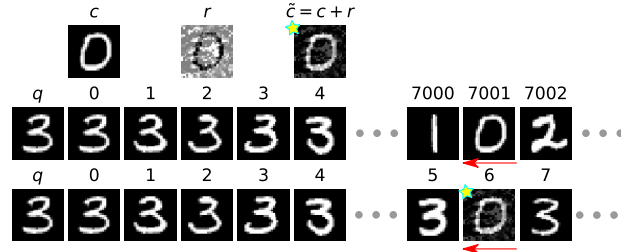


Figure 5. CA+ on MNIST. Example 3.

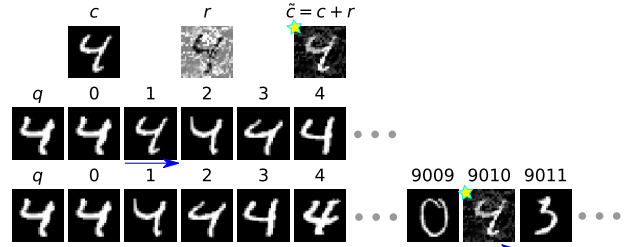


Figure 6. CA- on MNIST. Example 1.

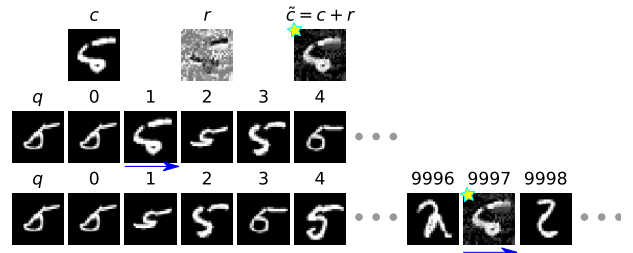


Figure 7. CA- on MNIST. Example 2.

of the Semantics-Preserving term in QA is substantial, as it keeps the retrieval results as “normal” as possible while achieving the attacking goal.

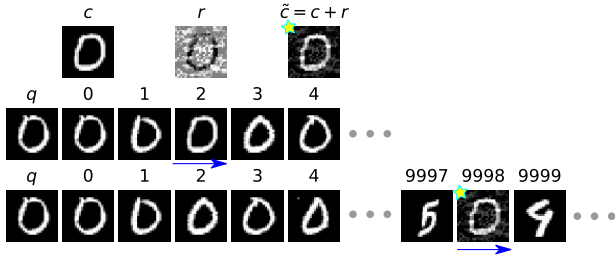


Figure 8. CA- on MNIST. Example 3.

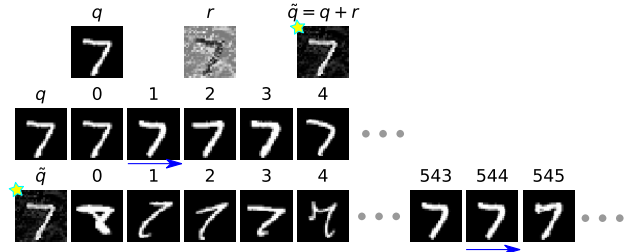


Figure 12. QA- on MNIST. Example 1.

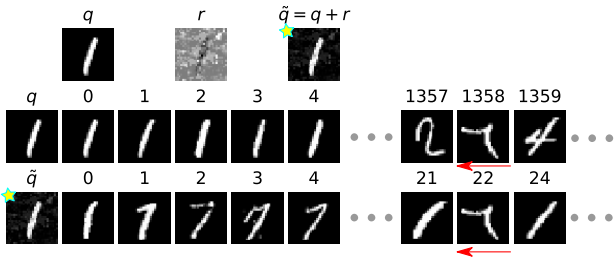


Figure 9. QA+ on MNIST. Example 1.

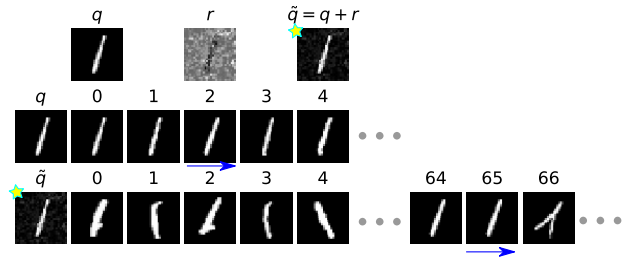


Figure 13. QA- on MNIST. Example 2.

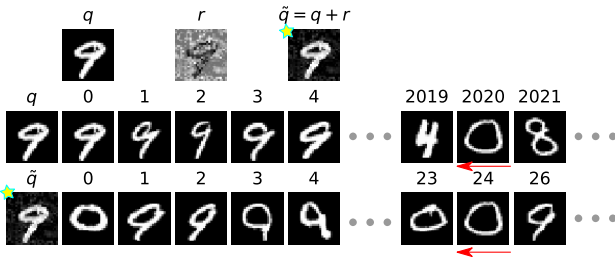


Figure 10. QA+ on MNIST. Example 2.

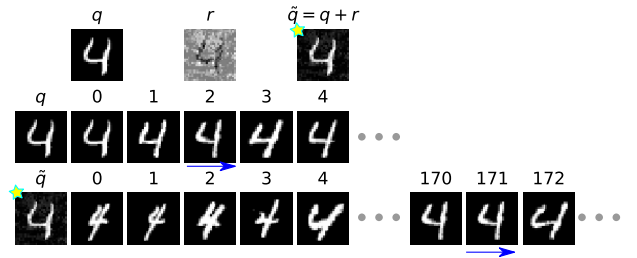


Figure 14. QA- on MNIST. Example 3.

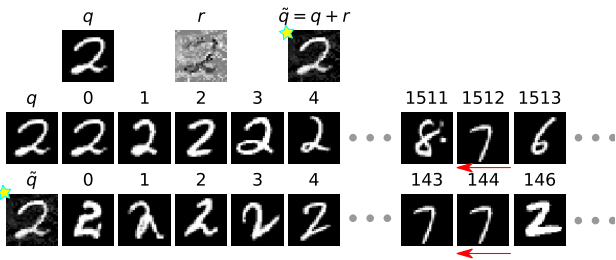


Figure 11. QA+ on MNIST. Example 3.

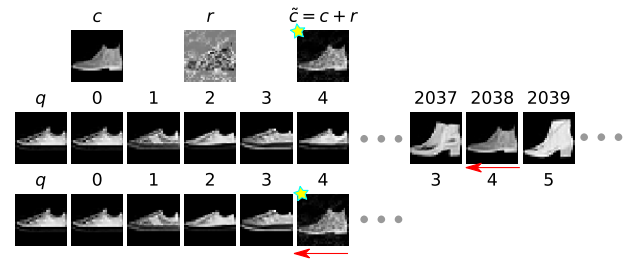


Figure 15. CA+ on Fashion-MNIST. Example 1.

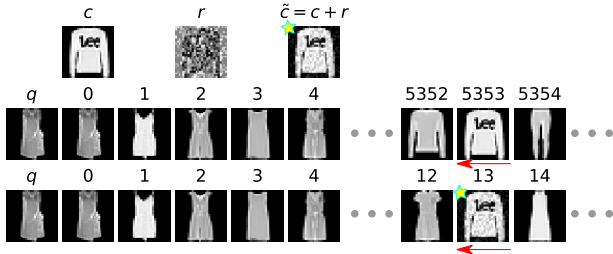


Figure 16. CA+ on Fashion-MNIST. Example 2.

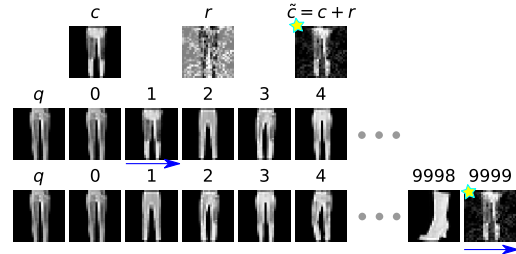


Figure 20. CA- on Fashion-MNIST. Example 3.

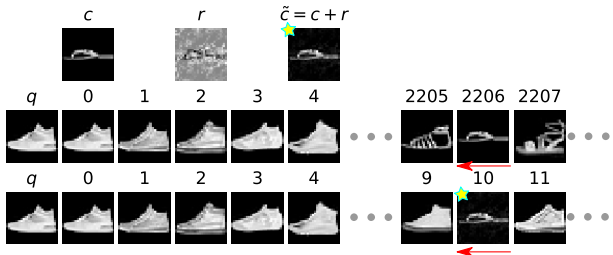


Figure 17. CA+ on Fashion-MNIST. Example 3.

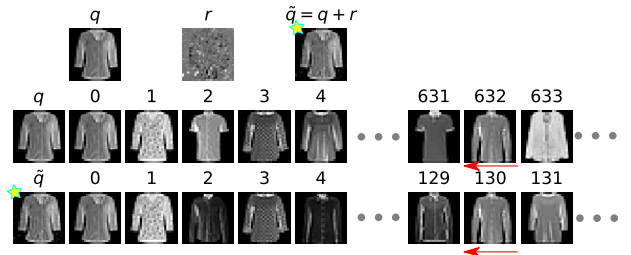


Figure 21. QA+ on Fashion-MNIST. Example 1.

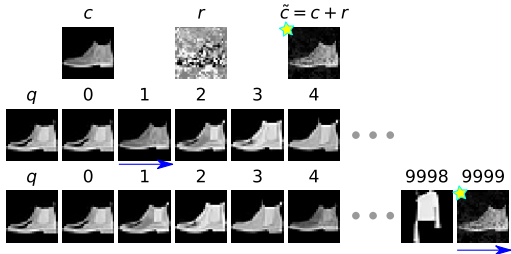


Figure 18. CA- on Fashion-MNIST. Example 1.

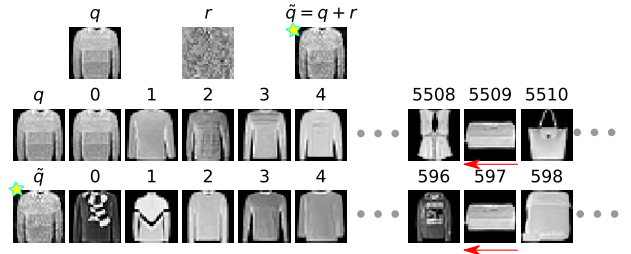


Figure 22. QA+ on Fashion-MNIST. Example 2.

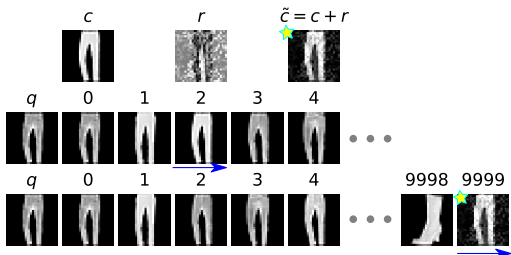


Figure 19. CA- on Fashion-MNIST. Example 2.

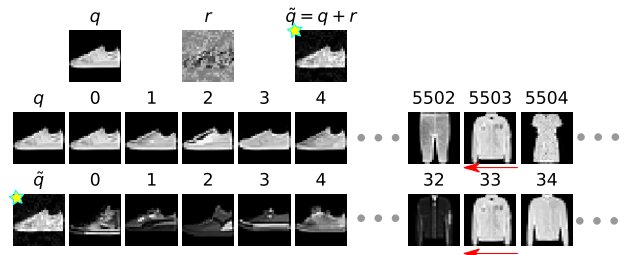


Figure 23. QA+ on Fashion-MNIST. Example 3.

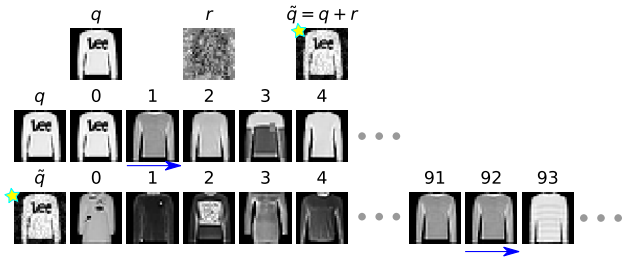


Figure 24. QA- on Fashion-MNIST. Example 1.

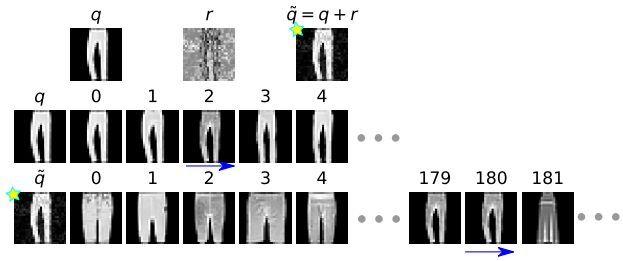


Figure 25. QA- on Fashion-MNIST. Example 2.

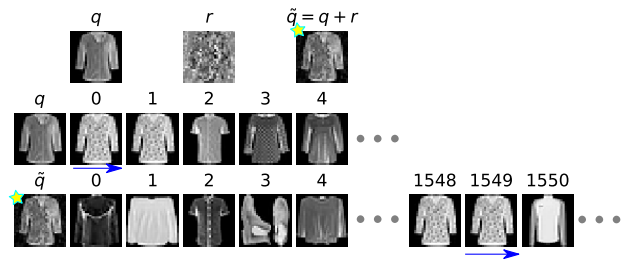


Figure 26. QA- on Fashion-MNIST. Example 3.

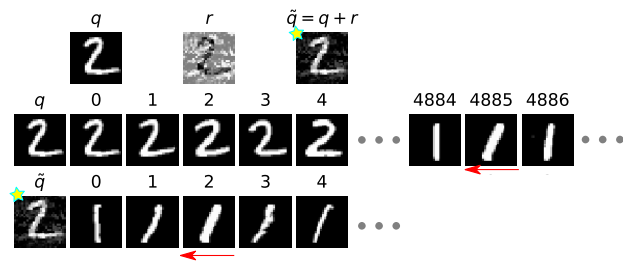


Figure 27. QA+ without preserving query semantics.

A.4. Stanford Online Products

CA+. See Fig. 28,29,30,31,32,33,34.

CA-. See Fig. 35,36,37,38,39,40,41.

QA+. See Fig. 42,43,44,45,46,47,48.

QA-. See Fig. 49,50,51,52,53,54,55.

CA+ ($w = 2$). See Fig. 56, 57, 58.

QA+ ($m = 2$). See Fig. 59, 60, 61.

CA+ ($w = 10$). See Fig. 62.

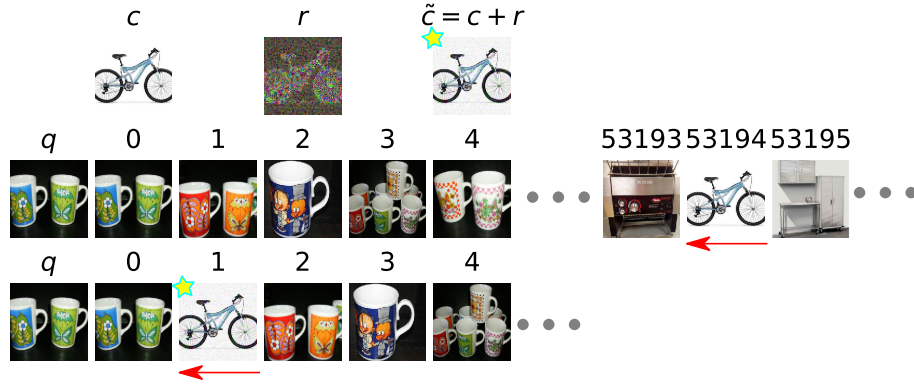


Figure 28. CA+ on SOP. Example 1.

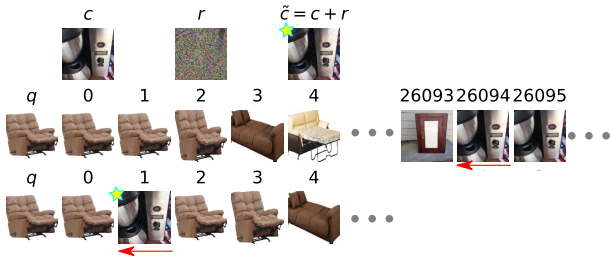


Figure 29. CA+ on SOP. Example 2.

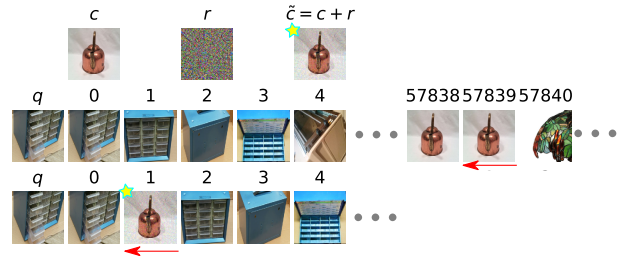


Figure 32. CA+ on SOP. Example 5.

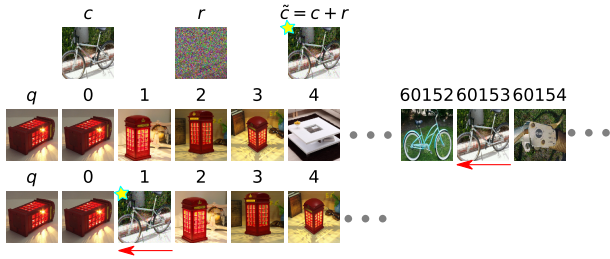


Figure 30. CA+ on SOP. Example 3.

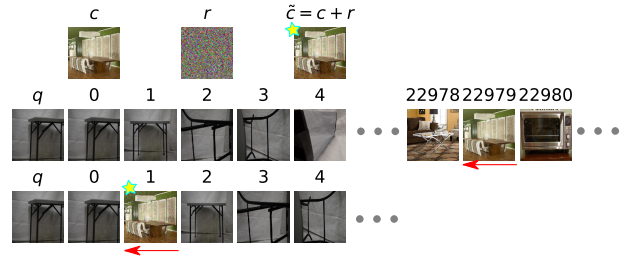


Figure 33. CA+ on SOP. Example 6.

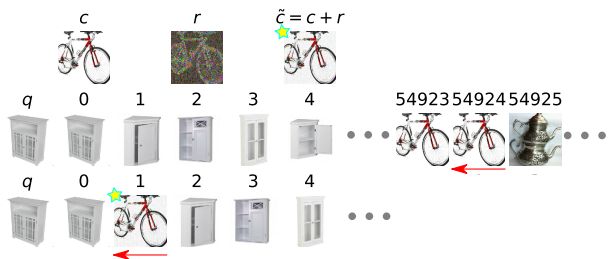


Figure 31. CA+ on SOP. Example 4.

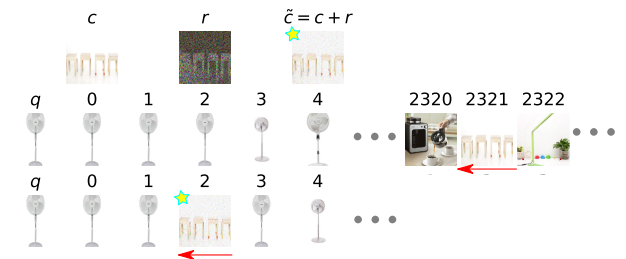


Figure 34. CA+ on SOP. Example 7.

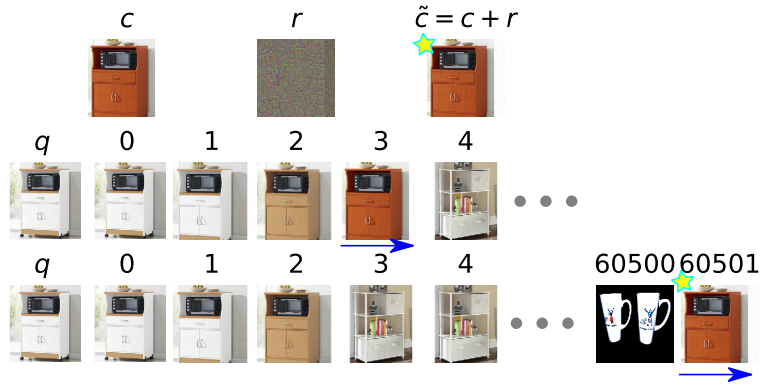


Figure 35. CA- on SOP. Example 1.

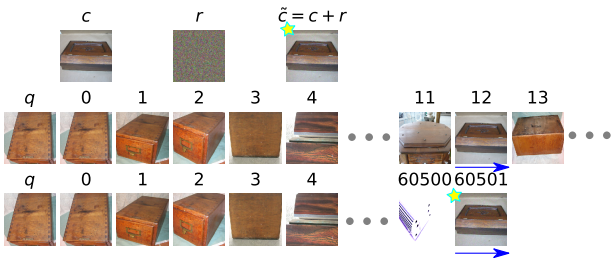


Figure 36. CA- on SOP. Example 2.

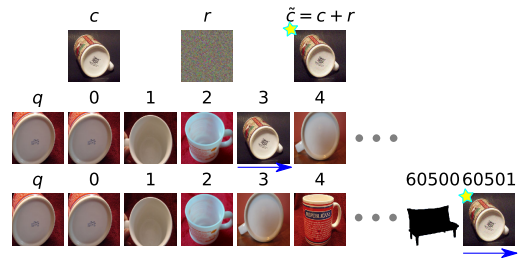


Figure 39. CA- on SOP. Example 5.

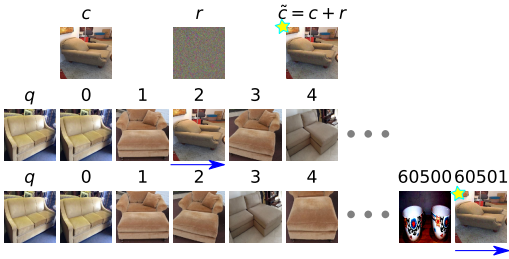


Figure 37. CA- on SOP. Example 3.

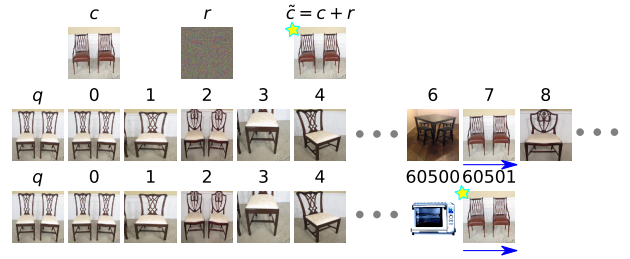


Figure 40. CA- on SOP. Example 6.

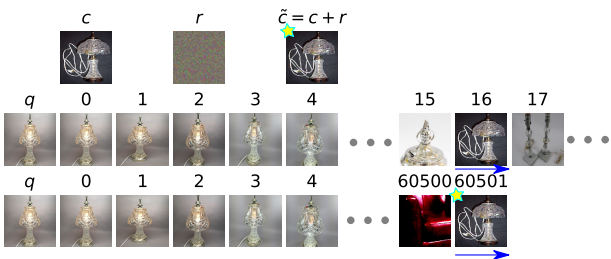


Figure 38. CA- on SOP. Example 4.

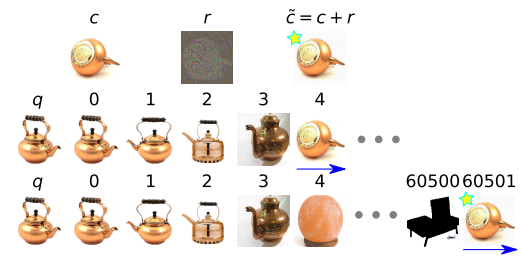


Figure 41. CA- on SOP. Example 7.

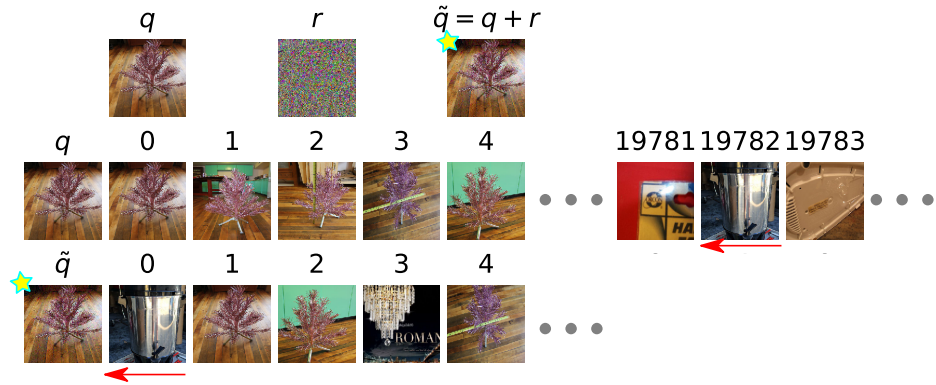


Figure 42. QA+ on SOP. Example 1.

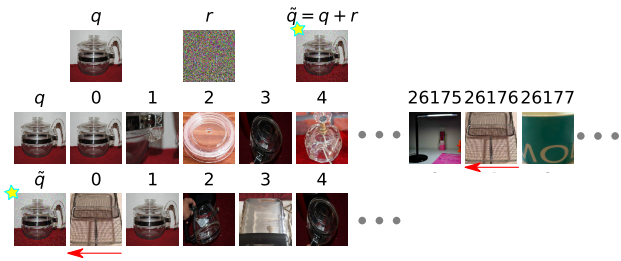


Figure 43. QA+ on SOP. Example 2.

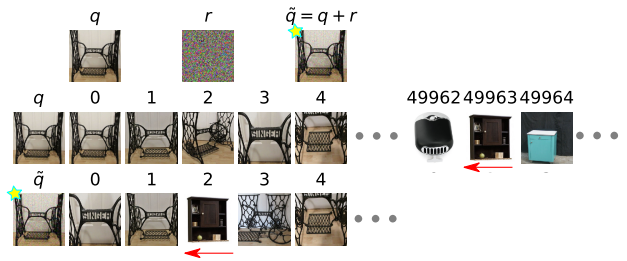


Figure 46. QA+ on SOP. Example 5.

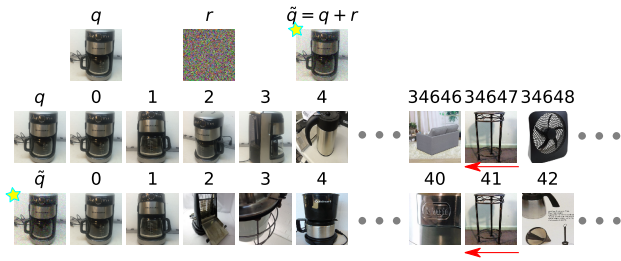


Figure 44. QA+ on SOP. Example 3.

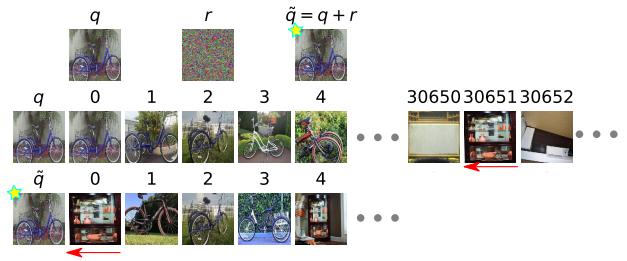


Figure 47. QA+ on SOP. Example 6.

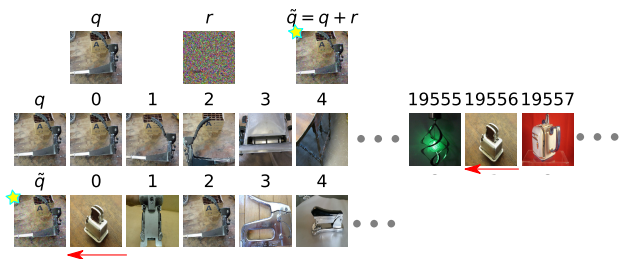


Figure 45. QA+ on SOP. Example 4.

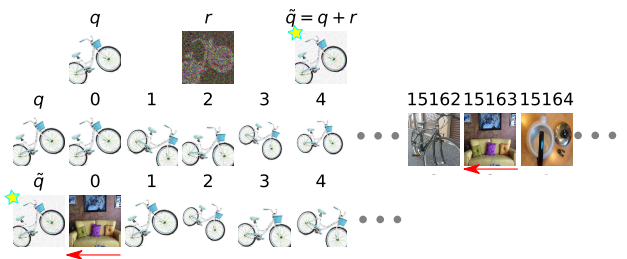


Figure 48. QA+ on SOP. Example 7.

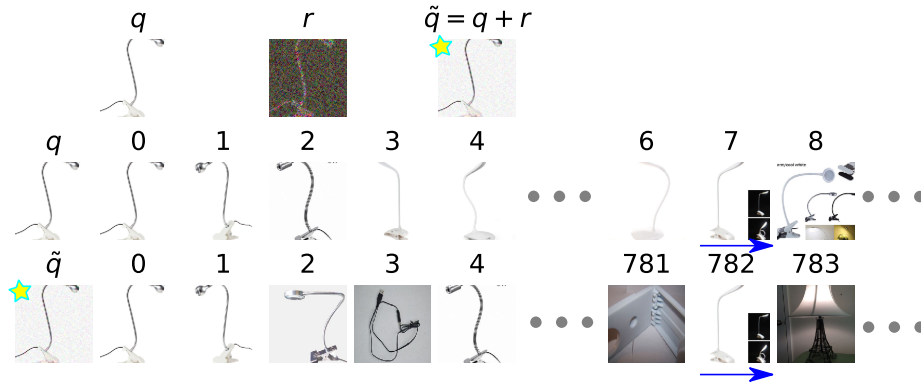


Figure 49. QA- on SOP. Example 1.

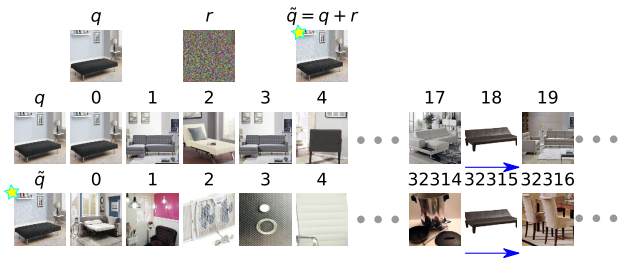


Figure 50. QA- on SOP. Example 2.

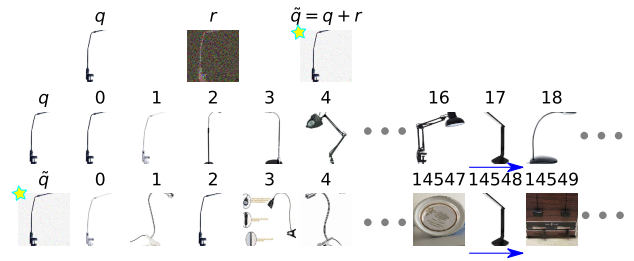


Figure 53. QA- on SOP. Example 5.

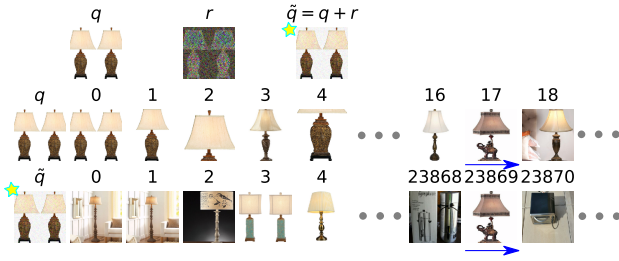


Figure 51. QA- on SOP. Example 3.

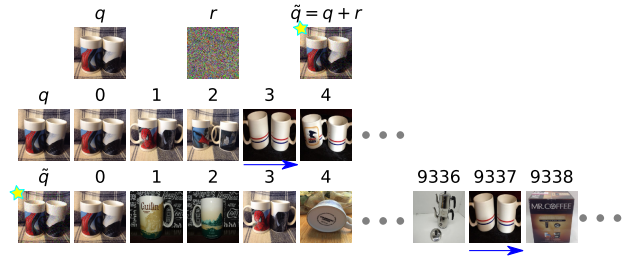


Figure 54. QA- on SOP. Example 6.

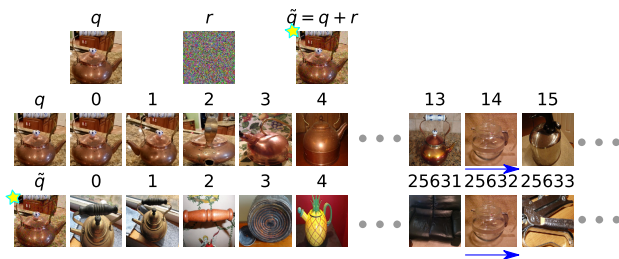


Figure 52. QA- on SOP. Example 4.

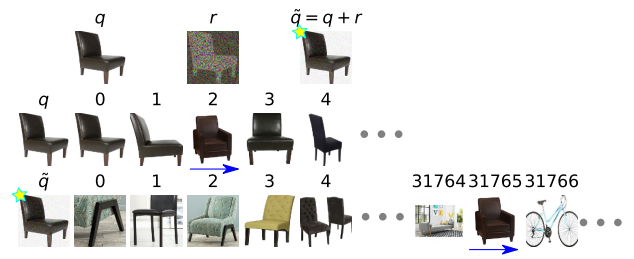


Figure 55. QA- on SOP. Example 7.

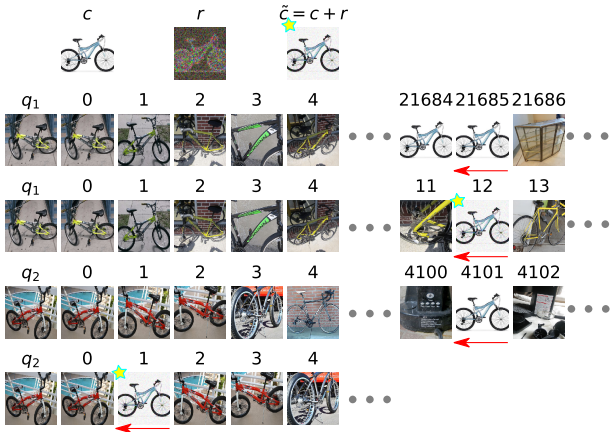


Figure 56. CA+ ($w = 2$) on SOP. Example 1.

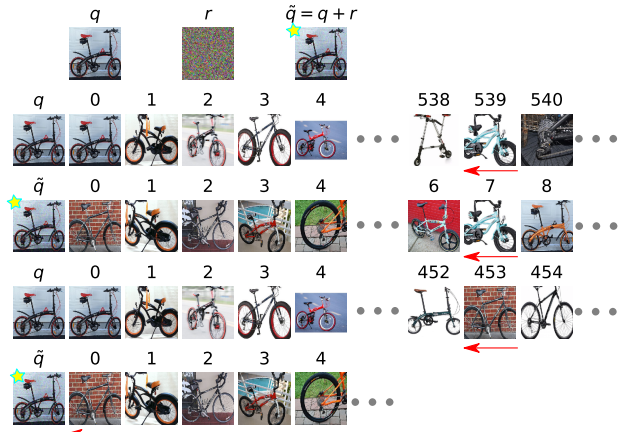


Figure 59. QA+ ($m = 2$) on SOP. Example 1.

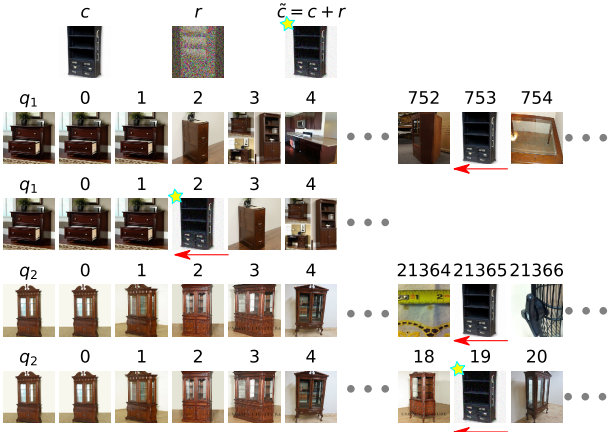


Figure 57. CA+ ($w = 2$) on SOP. Example 2.

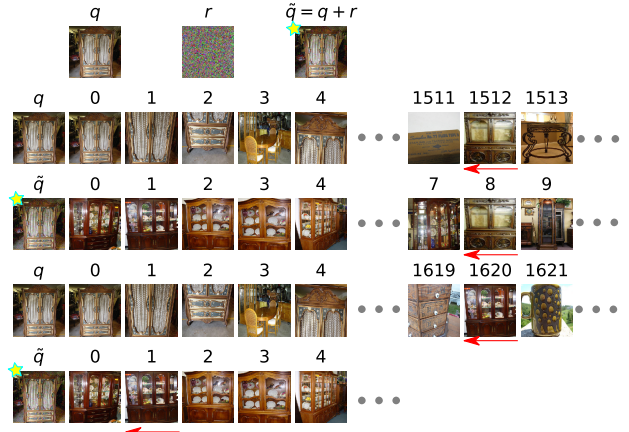


Figure 60. QA+ ($m = 2$) on SOP. Example 2.

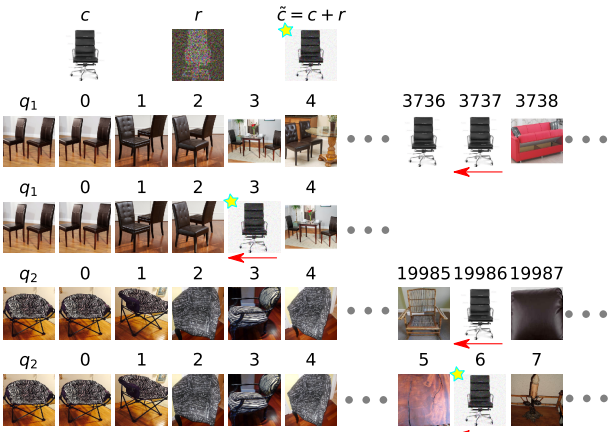


Figure 58. CA+ ($w = 2$) on SOP. Example 3.

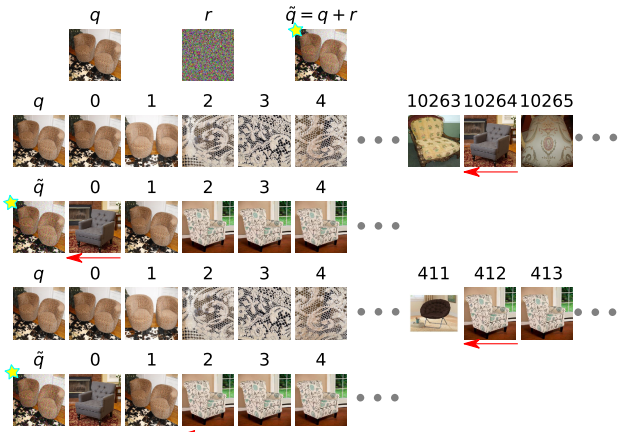


Figure 61. QA+ ($m = 2$) on SOP. Example 3.

B. Complete Results of Attack & Defense

Some experiments and details on MNIST, Fashion-MNIST and SOP datasets are omitted in the manuscript due to limited space. In this section, we present the complete experimental results on these datasets, including the average rank of C_{SP} during QA. Besides, we also conduct attacking experiments on ranking models trained with different combinations of loss functions and distance metrics. For brevity, we denote the (Cosine distance, Triplet loss) setting as (CT), the (Euclidean distance, Contrastive loss) as (EC). Models trained with our defense method will have a trailing “D” in notation, *e.g.* the (CT) model with our defense will be denoted as (CTD).

See Tab. 9 for complete results on MNIST.

See Tab. 10 for complete results on Fashion-MNIST.

See Tab. 8 for results on SOP.

ε	CA+					CA-					SP-QA+					SP-QA-				
	$w=1$	2	5	10		$w=1$	2	5	10		$m=1$	2	5	10		$m=1$	2	5	10	
(CC) Cosine Distance, Contrastive Loss (R@1=63.1%)																				
0	50	50	50	50		1.9	1.9	1.9	1.9		50	50	50	50		0.5	0.5	0.5	0.5	
0.01	0.0	0.8	2.0	2.6		99.7	99.6	99.3	99.3		4.8, 0.2	7.0, 0.7	16.3, 1.9	25.8, 2.3		54.9, 0.3	40.2, 0.6	27.1, 0.8	21.9, 0.9	
0.03	0.0	0.3	1.0	1.5		100.0	100.0	100.0	100.0		1.6, 0.1	3.3, 0.5	10.0, 2.0	19.2, 2.7		68.1, 0.3	52.4, 0.6	36.6, 0.8	30.1, 1.0	
0.06	0.0	0.2	1.0	1.5		100.0	100.0	100.0	100.0		1.1, 0.2	2.7, 0.6	8.8, 1.9	17.6, 3.3		73.8, 0.4	57.9, 0.7	40.3, 0.8	32.4, 1.0	
(ETD) Euclidean Distance, Triplet Loss, Defensive (R@1=46.4%)																				
0	50	50	50	50		2.0	2.0	2.0	2.0		50	50	50	50		0.5	0.5	0.5	0.5	
0.01	7.5	12.2	16.5	18.0		66.4	62.6	59.3	57.8		16.1, 2.1	24.8, 2.7	36.1, 2.7	41.4, 2.5		26.7, 0.6	18.1, 0.7	12.2, 0.7	10.2, 0.7	
0.03	0.7	4.5	8.7	10.4		91.7	90.2	89.1	88.4		7.9, 2.9	14.5, 4.2	27.2, 5.4	35.6, 5.3		43.4, 1.3	31.7, 1.5	21.9, 1.7	18.1, 1.8	
0.06	0.1	3.8	7.9	9.7		97.3	96.8	96.4	96.2		6.9, 4.3	12.5, 5.8	24.3, 7.0	33.4, 6.9		51.4, 2.8	39.0, 3.2	28.0, 3.5	23.5, 3.6	

Table 8. Experiments on SOP dataset.

ε	CA+					CA-					SP-QA+					SP-QA-					
	$w=1$	2	5	10		$w=1$	2	5	10		$m=1$	2	5	10		$m=1$	2	5	10		
(CC) Cosine Distance, Contrastive Loss (R@1=98.6%)																					
0	50	50	50	50		2.2	2.2	2.2	2.2		50	50	50	50		0.5	0.5	0.5	0.5		
0.01	42.2	43.3	44.6	44.7		5.6	5.3	5.2	5.1		45.0	46.9	0.5	48.4	0.5	49.1	0.5	12.0	12.0	11.0	11.0
0.03	31.3	33.5	35.4	36.1		9.3	9.2	9.1	9.1		37.6	41.7	2.5	45.9	2.5	47.7	2.5	2.8	1.8	2.7	1.8
0.1	9.7	12.5	14.6	15.6		52.1	52.2	52.4	52.5		18.5	8.2	27.1	9.7	38.0	9.9	43.4	9.9	5.6	4.1	5.4
0.3	5.8	10.0	11.9	12.9		99.0	99.1	99.0	99.1		14.3	9.7	23.3	12.5	36.3	13.8	42.6	13.9	6.1	4.4	5.4
(CTD) Cosine Distance, Triplet Loss (R@1=99.1%)																					
0	50	50	50	50		2.1	2.1	2.1	2.1		50	50	50	50		0.5	0.5	0.5	0.5		
0.01	44.6	45.4	47.4	47.9		3.4	3.2	3.1	3.1		45.2	0.0	46.3	0.0	47.7	0.0	48.5	0.0	0.9	0.0	0.7
0.03	33.4	37.3	41.9	43.9		6.3	5.9	5.7	5.6		35.6	0.3	39.2	0.3	43.4	0.3	45.8	0.3	1.9	0.2	1.4
0.1	12.7	17.4	24.4	30.0		15.4	14.9	14.8	14.7		14.4	2.2	21.0	2.3	30.6	2.3	37.2	2.3	5.6	1.2	4.4
0.3	2.1	9.1	13.0	17.9		93.9	93.2	93.0	92.9		6.3	3.6	11.2	5.7	22.5	7.7	32.1	7.7	8.6	1.6	6.6
(EC) Euclidean Distance, Contrastive Loss (R@1=99.0%)																					
0	50	50	50	50		1.8	1.8	1.8	1.8		50	50	50	50		0.5	0.5	0.5	0.5		
0.01	31.9	33.5	34.6	34.9		8.2	8.0	7.9	7.8		41.3	2.0	44.0	2.2	47.2	2.4	48.4	2.5	2.6	1.3	2.4
0.03	15.8	17.4	18.7	19.1		10.7	10.6	10.5	10.5		27.2	4.8	34.6	5.4	42.0	5.6	45.6	5.8	4.1	2.5	3.9
0.1	6.7	10.0	12.1	12.7		82.1	81.6	81.7	82.0		15.7	9.8	25.4	12.0	37.7	13.1	43.2	13.1	5.6	3.2	5.3
0.3	4.8	9.9	12.1	12.7		99.8	99.8	99.8	99.8		14.4	9.4	23.9	12.5	36.7	13.6	42.6	13.8	5.9	3.2	5.4
(ETD) Euclidean Distance, Triplet Loss (R@1=99.2%)																					
0	50	50	50	50		1.6	1.6	1.6	1.6		50	50	50	50		0.5	0.5	0.5	0.5		
0.01	39.0	40.4	40.6	40.8		3.2	3.0	2.8	2.8		45.5	0.1	47.4	0.1	48.1	0.1	49.0	0.1	1.0	0.1	0.9
0.03	23.5	25.6	26.5	27.0		7.0	6.6	6.3	6.2		36.4	0.6	40.9	0.7	45.1	0.8	47.1	0.9	2.7	0.7	2.3
0.1	8.1	10.6	12.1	12.7		13.7	13.3	13.0	12.9		14.1	4.8	23.4	5.7	34.6	6.2	40.8	6.5	6.2	1.7	5.1
0.3	1.8	8.5	10.9	11.7		77.4	75.7	75.0	74.7		7.5	5.2	15.0	7.4	28.1	9.0	36.3	8.9	7.4	1.7	5.8
(CCD) Cosine Distance, Contrastive Loss, Defensive (R@1=97.5%)																					
0	50	50	50	50		2.2	2.2	2.2	2.2		50	50	50	50		0.5	0.5	0.5	0.5		
0.01	49.3	49.0	49.3	49.3		2.4	2.4	2.3	2.3		49.7	0.0	49.6	0.0	49.8	0.0	49.9	0.0	0.5	0.0	0.5
0.03	47.0	47.9	48.1	48.0		2.8	2.7	2.7	2.7		48.3	0.0	49.1	0.0	49.1	0.0	49.4	0.0	0.6	0.0	0.5
0.1	42.3	43.3	44.0	44.4		4.2	4.0	3.9	3.8		44.9	0.1	46.7	0.1	48.1	0.1	48.8	0.1	0.8	0.1	0.7
0.3	32.0	34.2	36.1	36.7		7.0	7.0	6.5	6.4		37.4	0.6	41.1	0.6	44.9	0.5	47.2	0.5	1.9	0.5	1.6
(CTD) Cosine Distance, Triplet Loss, Defensive (R@1=98.3%)																					
0	50	50	50	50		2.0	2.0	2.0	2.0		50	50	50	50		0.5	0.5	0.5	0.5		
0.01	48.9	49.3	49.4	49.5		2.2	2.2	2.2	2.1		49.9	0.0	49.5	0.0	49.5	0.0	49.7	0.0	0.5	0.0	0.5
0.03	47.4	48.4	48.6	48.9		2.5	2.5	2.4	2.4		48.0	0.0	48.5	0.0	49.2	0.0	49.5	0.0	0.6	0.0	0.5
0.1	42.4	44.2	45.9	46.7		3.8	3.6	3.5	3.4		43.2	0.1	45.0	0.1	47.4	0.1	48.2	0.1	1.0	0.1	0.8
0.3	30.7	34.5	38.7	40.7		7.0	6.7	6.5	6.5		33.2	0.5	37.2	0.5	42.3	0.5	45.1	0.5	2.4	0.4	1.9
(ECD) Euclidean Distance, Contrastive Loss, Defensive (R@1=97.9%)																					
0	50	50	50	50		1.3	1.3	1.3	1.3		50	50	50	50		0.5	0.5	0.5	0.5		
0.01	47.4	47.6	48.0	47.8		2.3	2.2	2.1	2.1		48.7	0.1	49.1	0.1	49.4	0.1	49.7	0.1	0.9	0.1	0.8
0.03	42.7	43.6	44.0	44.2		4.5	4.2	4.0	4.0		46.3	0.6	48.1	0.6	48.8	0.6	49.2	0.6	1.8	0.6	1.6
0.1	31.7	33.7	34.9	35.3		10.0	9.6	9.4	9.3		39.2	2.8	43.2	2.9	46.6	2.9	47.9	2.9	3.3	1.3	2.9
0.3	19.6	23.0	25.4	26.3		35.6	35.2	35.7	36.0		27.3	7.1	34.4	7.4	42.2	7.1	45.4	6.9	4.5	1.8	3.9
(ETD) Euclidean Distance, Triplet Loss, Defensive (R@1=99.0%)																					
0	50	50	50	50		1.4	1.4	1.4	1.4		50	50	50	50		0.5	0.5	0.5	0.5		
0.01	47.5	48.2	48.1	48.2		1.7	1.7	1.6	1.6		48.5	0.0	48.8	0.0	49.4	0.0	49.7	0.0	0.6	0.0	0.6
0.03	43.4	43.8	44.1	44.5		2.4	2.3	2.2	2.2		46.6	0.0	47.8	0.0	48.9	0.1	49.3	0.1	0.8	0.1	0.7
0.1	29.8	31.3	32.6	33.0		5.7	5.2	5.0	4.9		39.6	0.3	42.9	0.3	45.9	0.4	47.9	0.4	2.1	0.4	1.7
0.3	10.8	13.2	15.0	15.6		14.4	13.9	13.5	13.4		29.7	2.8	33.1	2.7	37.4	2.4	42.6	2.3	6.5	1.7	5.3

C. Complete Results on Transferability

C.1. Fashion-MNIST

In addition to the transferability experiment on MNIST dataset, we also conduct the same transferability experiment on the Fashion-MNIST dataset, as shown in Tab. 11.

CA+ Transfer (Black Box), $w = 1$				
From \ To	LeNet	C2F1	Res18	
LeNet	50→16.0	41.0	44.3	
C2F1	38.6	50→1.3	40.3	
Res18	39.2	34.3	50→1.7	
CA- Transfer (Black Box), $w = 1$				
From \ To	LeNet	C2F1	Res18	
LeNet	2.5→84.3	1.9→8.1	1.6→6.0	
C2F1	2.5→7.8	1.9→100.0	1.7→7.7	
Res18	2.5→9.5	1.9→14.4	1.7→80.0	
SP-QA+ Transfer (Black Box), $m = 1$				
From \ To	LeNet	C2F1	Res18	
LeNet	50→18.0	47.2	49.3	
C2F1	48.1	50→6.4	49.2	
Res18	48.1	44.8	50→13.7	
SP-QA- Transfer (Black Box), $m = 1$				
From \ To	LeNet	C2F1	Res18	
LeNet	0.5→13.5	0.5→1.7	0.5→1.5	
C2F1	0.5→1.1	0.5→12.5	0.5→1.6	
Res18	0.5→0.9	0.5→1.3	0.5→8.0	

Table 11. Transferability experiment on Fashion-MNIST dataset.

C.2. “Self-Transfer” Attack on MNIST

In the adversarial ranking example transferability experiments, we transfer adversarial examples between neural networks with different architectures. In this section, we transfer adversarial examples between neural networks based on the same architecture but with different parameters.

We train three vanilla C2F1 models (denoted as C2F1-1, C2F1-2, and C2f1-3) on MNIST dataset and two models with our defense (denoted as C2F1-D1, C2F1-D2). All these models have exactly the same architecture, but different parameters. Transferability experimental results between these models are present in Tab. 12.

D. Complete Results for Universal Perturbation

D.1. MNIST

See Tab. 13.

D.2. Fashion-MNIST

As shown in Tab. 14, Image-agnostic adversarial perturbation has better effect on seen data from Fashion-MNIST,

CA+ Transfer (Black Box), $w = 1$					
From \ To	C2F1-1	C2F1-2	C2F1-3	C2F1-D1	C2F1-D2
C2F1-1	50→2.1	27.9	25.3	45.3	45.1
C2F1-2	24.4	50→2.7	23.7	44.5	44.0
C2F1-3	22.6	24.4	50→2.2	45.0	44.5
C2F1-D1	36.7	37.7	37.4	50→30.9	38.6
C2F1-D2	37.3	36.8	37.3	39.6	50→30.3
CA- Transfer (Black Box), $w = 1$					
From \ To	C2F1-1	C2F1-2	C2F1-3	C2F1-D1	C2F1-D2
C2F1-1	2.1→93.7	2.2→9.4	2.1→11.1	2.0→3.2	1.9→3.1
C2F1-2	2.1→13.5	2.1→88.1	2.2→12.3	2.0→3.4	1.9→3.3
C2F1-3	2.1→13.4	2.2→10.1	2.2→92.4	2.0→3.2	1.9→3.2
C2F1-D1	2.1→9.3	2.2→8.3	2.1→9.1	2.0→7.0	2.0→4.9
C2F1-D2	2.1→9.2	2.2→8.4	2.1→9.1	2.0→4.9	1.9→7.0
SP-QA+ Transfer (Black Box), $m = 1$					
From \ To	C2F1-1	C2F1-2	C2F1-3	C2F1-D1	C2F1-D2
C2F1-1	50→6.4	39.4	38.1	48.6	48.1
C2F1-2	36.6	50→6.3	36.4	47.3	48.4
C2F1-3	36.9	37.8	50→6.2	48.7	48.3
C2F1-D1	41.5	40.3	41.4	50→32.9	40.6
C2F1-D2	41.7	38.9	41.7	41.4	50→32.7
SP-QA- Transfer (Black Box), $m = 1$					
From \ To	C2F1-1	C2F1-2	C2F1-3	C2F1-D1	C2F1-D2
C2F1-1	0.5→8.8	0.5→1.4	0.5→1.9	0.5→0.6	0.5→0.6
C2F1-2	0.5→2.3	0.5→9.0	0.5→2.2	0.5→0.6	0.5→0.6
C2F1-3	0.5→1.9	0.5→1.5	0.5→8.7	0.5→0.6	0.5→0.6
C2F1-D1	0.5→5.8	0.5→3.1	0.5→4.8	0.5→2.4	0.5→1.3
C2F1-D2	0.5→7.1	0.5→3.7	0.5→5.9	0.5→1.3	0.5→2.4

Table 12. Transfer experiment between models based on the same architecture but with different parameters.

Model	I-CA+ ($w = 1$)		I-CA- ($w = 1$)		I-QA+ ($m = 1$)		I-QA- ($m = 1$)	
	Seen	Unseen	Seen	Unseen	Seen	Unseen	Seen	Unseen
(CC)	50 → 13.7	50 → 14.1	0.6 → 37.6	0.6 → 33.4	50 → 18.1	50 → 18.9	2.4 → 41.6	2.4 → 39.3
(CT)	50 → 18.1	50 → 18.5	0.6 → 9.5	0.7 → 9.4	50 → 20.5	50 → 21.0	2.1 → 7.6	2.2 → 7.4
(EC)	50 → 9.1	50 → 10.3	1.9 → 94.6	2.0 → 79.4	50 → 3.9	50 → 6.9	3.3 → 87.1	3.5 → 74.4
(ET)	50 → 11.5	50 → 12.6	2.1 → 10.6	2.1 → 9.6	50 → 21.6	50 → 23.6	3.2 → 28.6	3.2 → 19.5

Table 13. Universal Adversarial Ranking Perturbation on MNIST. Each pair of result presents the original rank of chosen candidate(s), and the rank of the chosen candidate(s) after adding adversarial perturbation to the candidate or to the query. “Seen” samples are those used for generating the universal perturbation, while “Unseen” samples are a set of other non-overlapping samples.

Model	I-CA+ ($w = 1$)		I-CA- ($w = 1$)		I-QA+ ($m = 1$)		I-QA- ($m = 1$)	
	Seen	Unseen	Seen	Unseen	Seen	Unseen	Seen	Unseen
(CC)	50 → 7.0	50 → 7.3	0.6 → 91.8	0.6 → 84.9	50 → 4.4	50 → 4.9	2.1 → 87.5	2.1 → 84.4
(CT)	50 → 9.8	50 → 9.9	0.6 → 72.3	0.6 → 69.7	50 → 8.2	50 → 8.4	2.0 → 76.3	2.0 → 72.9
(EC)	50 → 5.8	50 → 9.6	2.0 → 97.5	1.9 → 83.7	50 → 1.8	50 → 7.1	2.9 → 87.9	2.8 → 78.4
(ET)	50 → 5.7	50 → 8.5	2.0 → 84.4	1.9 → 69.9	50 → 3.3	50 → 6.3	3.1 → 88.0	3.0 → 78.0

Table 14. Image-agnostic Adversarial Perturbation on Fashion-MNIST.

but the gap between the effect on seen samples and that on unseen samples is slightly larger, which may due to the higher intra-class variance of Fashion-MNIST than MNIST.

ξ	SP-QA+				SP-QA-			
	m=1	2	5	10	m=1	2	5	10
(CC) Cosine, Contrastive								
0	1.4, 43.9	13.8, 38.1	30.7, 37.9	39.4, 36.3	94.7, 93.3	94.4, 93.5	94.2, 93.7	94.1, 93.8
10^0	14.3, 9.7	23.3, 12.5	36.3, 13.8	42.6, 13.9	40.6, 37.6	40.6, 38.1	40.4, 38.1	39.8, 37.7
10^2	25.4, 2.0	33.2, 2.1	41.7, 2.1	45.4, 2.1	6.1, 4.4	5.8, 4.4	5.7, 4.4	5.7, 4.4
10^4	48.1, 0.3	49.2, 0.2	49.6, 0.2	49.8, 0.3	1.2, 0.3	1.1, 0.3	1.1, 0.3	1.1, 0.3
(CT) Cosine, Triplet								
0	0.2, 33.6	6.3, 23.7	18.5, 26.5	29.6, 25.7	94.1, 89.4	93.2, 90.3	92.6, 90.9	92.3, 91.2
10^0	6.3, 3.6	11.2, 5.7	22.5, 7.7	32.1, 7.7	55.5, 35.6	52.4, 37.6	50.2, 39.3	49.4, 40.0
10^2	14.1, 0.6	20.8, 0.7	31.2, 0.7	38.1, 0.7	8.6, 1.6	6.6, 1.6	5.3, 1.5	4.8, 1.5
10^4	37.9, 0.1	42.6, 0.1	46.3, 0.1	47.8, 0.1	1.9, 0.1	1.4, 0.1	1.2, 0.1	1.1, 0.1
(EC) Euclidean, Contrastive								
0	0.7, 44.5	13.4, 39.9	31.0, 41.8	39.7, 41.6	94.0, 92.6	93.9, 93.0	93.9, 93.4	93.9, 93.5
10^0	14.4, 9.4	23.9, 12.5	36.7, 13.6	42.6, 13.8	39.9, 37.6	39.0, 37.2	39.2, 37.7	38.7, 37.3
10^2	30.1, 1.2	37.3, 1.1	43.9, 0.9	46.7, 0.9	5.9, 3.2	5.4, 3.2	5.0, 3.2	4.9, 3.1
10^4	50.1, 0.8	50.2, 0.8	49.8, 0.8	50.0, 0.8	1.6, 0.7	1.6, 0.7	1.7, 0.8	1.6, 0.8
(ET) Euclidean, Triplet								
0	0.6, 37.4	8.2, 35.1	22.6, 38.2	33.2, 38.8	93.0, 88.4	92.5, 89.9	92.3, 90.9	92.1, 90.9
10^0	7.5, 5.2	15.0, 7.4	28.1, 9.0	36.3, 8.9	50.9, 37.3	49.2, 39.4	47.7, 40.6	47.2, 40.9
10^2	20.3, 0.5	28.6, 0.5	38.2, 0.5	43.1, 0.4	7.4, 1.7	5.8, 1.7	4.9, 1.7	4.6, 1.7
10^4	46.2, 0.1	48.2, 0.1	49.1, 0.1	49.6, 0.1	1.7, 0.1	1.4, 0.1	1.2, 0.1	1.1, 0.1

Table 15. Parameter search on Semantics-Preserving balancing parameter ξ with MNIST. We report two mean ranks in each cell: one for the chosen candidate(s) C , the other for the C_{SP} used for SP.

E. Complete Parameter Search on ξ

See Tab. 15.

F. Analysis on Our Defense

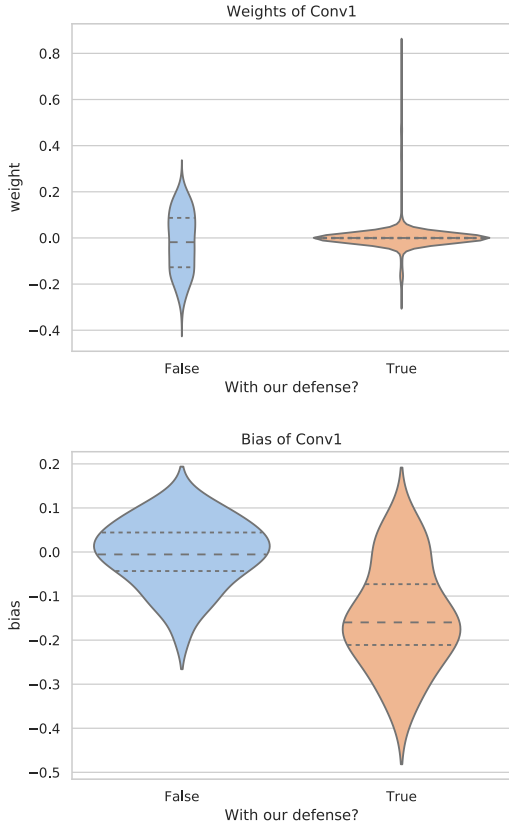


Figure 63. Weights and bias of the first convolutional layer (conv1) with/without our defense.

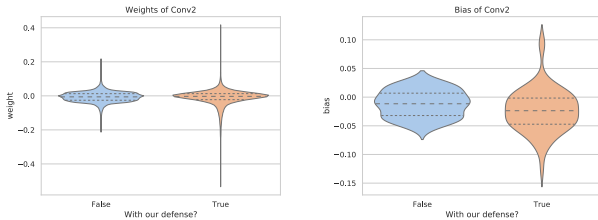


Figure 64. Weights and bias of the second convolutional layer (conv2) with/without our defense.

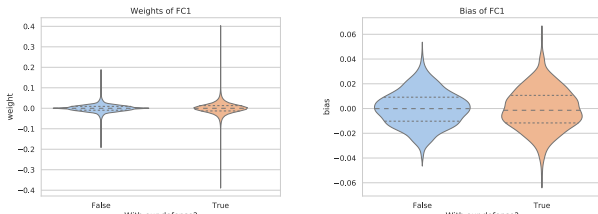


Figure 65. Weights and bias of the first fully-connected layer (fc1) with/without our defense.

Ian *et al.* [21] proposed a linear explanation of adversarial examples:

Consider the dot product between a weight vector w and an adversarial example $\tilde{x} = x + r$ ($\|r\|_\infty < \varepsilon$):

$$w^T \tilde{x} = w^T x + w^T r. \quad (15)$$

The adversarial perturbation causes the activation to grow by $w^T r$. We can maximize this increase subject to the max norm constraint on r by assigning $r = \varepsilon \text{sign}(w)$. If w has k dimensions and the average magnitude of an element of the weight vector is h , then the activation will grow by $\varepsilon k h$. Since $\|r\|_\infty$ does not grow with the dimensionality of the problem but the change in activation caused by perturbation by r can grow linearly with k , then for high dimensional problems, we can make many infinitesimal changes to the input that add up to one large change to the output.

In our experiments, we analysed the parameters in models trained on MNIST, as shown in Fig. 63, 64, 65. These violin plots suggest that

- the weights in the first convolution layer of the defensive model are closer to 0 and have smaller variance than those of the vanilla model. That means the h in the above quotation is decreased with our defense, and it will be harder for the r to incur a large increase in activation $\varepsilon k h$.
- the bias in the first convolution layer of the defensive model tend to be negative values instead of being nearly “zero-mean”. These negative bias could help further suppress the increase in activation caused by perturbation r .

Therefore, Ian *et al.*’s [21] theory could explain why our defense works, as the resulting differences in network parameters may help reduce the probability for adversarial perturbation to increase the layer outputs into the local linear area of ReLU.

G. Alternative Attack

G.1. Distance-based Ranking Attack

In order to implement the proposed ranking attack, alternative attacking objectives are possible. Some related works such as **Feature Adversary** [66] generates untargeted adversarial examples against classifiers by maximizing the distance shift of representation vectors off their original locations. This may inspire an alternative version of CA or QA objective functions which are directly based on distance. For example, such alternative objective for CA+ and QA- could be as follows:

$$r = \arg \min_{r \in \Gamma} \sum_{q \in Q} d(q, c + r) \quad (16)$$

$$r = \arg \max_{r \in \Gamma} \sum_{c \in C} d(q + r, c). \quad (17)$$

However, it must be pointed out that our method significantly differs from feature adversary [66]: (1) Feature adversary concerns the *pairwise* similarity of source-target representations, while our image ranking problem concerns the ranking order of multiple candidates; (2) Feature adversary attempts to reduce the ℓ_2 distance as much as possible, while our triplet-like loss attempts to make positive candidates closer to query than the negative ones, which well fits the objective of ranking order optimization. Such *relative* distance optimization becomes more important since our attack simultaneously involves multiple queries and multiple candidates; (3) The ℓ_2 distance based methods suffer from inevitable disadvantages. Specifically, distance-based objectives are suboptimal, because they disregard the relative positions among the candidates and queries.

As shown in the top-left part of Fig. 66, the solution set for for distance-based CA+ (the green dotted line) contains suboptimal solutions. Similarly, in the bottom-left part of Fig. 66 the distance-based objective for QA- tends to maximize the sum of distance neglecting the ranking result, and further optimization (moving \tilde{q} along the green arrow) will not change the ranking result. In contrast, our proposed inequality-based method does not suffer from these issues, as shown in the top-right and bottom-right parts.

We also implemented such distance-based method and compared it with our triplet-like method, as shown in Tab. 16. Experimental results suggest that our method always outperforms distance-based method by a margin. Especially for QA-, the distance-based objective is very difficult to optimize because the distance-based Semantics-Preserving term contradicts with the other term in the loss function. In summary, distance-based method is not well-suited for our proposed *adversarial ranking attack*, especially in the scenario of QA-.

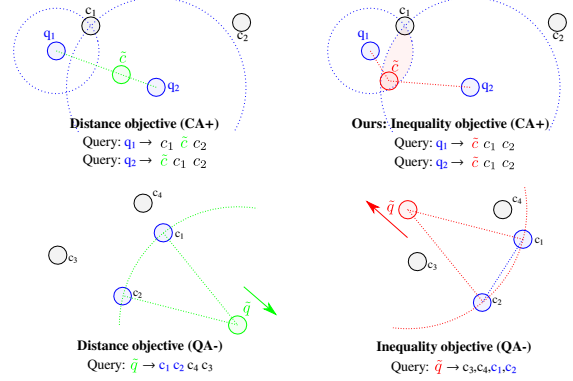


Figure 66. Distance-based CA+ (top-left) and QA- (bottom-left) objective functions do not lead to desired attacking result compared to our inequality-based method (top-right and bottom-right). In the top-left diagram for CA+, q_1 and q_2 are the chosen queries, while \tilde{c} is the adversarial candidate found by the distance objective. In the bottom-left diagram of QA-, c_1 and c_2 are the chosen candidates, while \tilde{q} is the adversarial query found by the distance objective. It is noted that the distance objective is not optimal as shown in the top-left part, where the solution set for minimizing the distance objective contains suboptimal results. Distance-based objective may even fail to change the ranking result as shown in the bottom-left part, where optimizing the objective further cannot change the ranking result. In contrast, our proposed inequality-based method does not suffer from these issues, as shown in the top-right and bottom-right parts.

ε	CA+					CA-					QA+					QA-				
	$w=1$	2	5	10		$w=1$	2	5	10		$m=1$	2	5	10		$m=1$	2	5	10	
0.3	3.0	9.5	15.9	22.2		86.0	85.2	84.7	84.6		7.4	20.2	34.9	41.7		0.8	0.8	0.8	0.8	

Table 16. Attack Based on L-2 Distance Loss with MNIST.

H. Alternative Defense

Apart from the defense provided in the manuscript, we also tried some other loss functions for adversarial training. In literature, there is no predominant choice for the adversarial training loss function. The only common trait among these choices is that all of them involve adversarial examples. Inspired by previous works, we also implement some alternative defenses for ranking systems as follows.

H.1. Straightforward Adaptation of Madry Defense

Madry [50] formularised improving neural network classifier robustness as a min-max optimization problem, where the inner maximization seeks to generate adversarial examples that lead to maximum cross-entropy loss L_{CE} , while the outer minimization tunes the neural network parameters θ to suppress the cross-entropy loss:

$$\min_{\theta} \left\{ \mathbb{E}_{(x,y) \sim D} \left[\max_{r \in \Gamma} L_{CE}(x + r, y) \right] \right\} \quad (18)$$

where (x, y) is a pair of image and ground-truth class label.

Similarly, we follow the idea and use a similar defense for ranking models:

$$\min_{\theta} \left\{ \mathbb{E}_{(q, c_p, c_n) \sim D} \left[\max_{r \in \Gamma} L_{\text{triplet}}(q + r, c_p, c_n) \right] \right\} \quad (19)$$

where the inner maximization aims to generate strongest adversarial examples that could lead to triplet ranking error and a large loss value, while the outer minimization seeks network parameters that could reduce such error.

However, during experiments, we observe that such defensive loss function always diverges, possibly due to the adversarial examples generated by the inner problem being too “strong”. We leave further investigation into this problem for future work.

H.2. Straightforward Adaptation of Ian Defense

Ian [21] proposed the following loss function for adversarial training:

$$L_{\text{Ian}}(x, y) = \alpha L_{CE}(x, y) \quad (20)$$

$$+ (1 - \alpha) L_{CE}(x + \varepsilon \text{sign}(\nabla_x L_{CE}(x, y))) \quad (21)$$

where the first term is a normal Cross-Entropy loss, and the second term is the cross-entropy loss with untargeted adversarial example that aims to increase the loss value. Constant α is a balancing parameter.

When adapted to a deep ranking system, this defense method also suffers from the diverging issue similar to Madry’s defense.

H.3. Directly Suppressing Shift Distance

As discussed in the manuscript, another possible adversarial training method could be to directly suppress the max-

ε	CA+					CA-					QA+					QA-					
	w=1	2	5	10		w=1	2	5	10		m=1	2	5	10		m=1	2	5	10		
(CCD) Cosine Distance, Contrastive Loss, Defensive (R@1=95.3%)																					
0	50	50	50	50		1.2	1.2	1.2	1.2		50	50	50	50		0.5	0.5	0.5	0.5		
0.01	49.1	49.3	49.6	49.5		1.3	1.3	1.3	1.3		49.7	0.0	49.9	0.0	50.0	0.0	49.8	0.0	0.5	0.0	0.5
0.03	48.0	48.2	48.5	48.6		1.6	1.5	1.5	1.5		48.7	0.0	49.2	0.0	49.7	0.0	49.7	0.0	0.6	0.0	0.5
0.1	43.1	44.4	45.2	45.5		2.4	2.3	2.1	2.1		45.3	0.1	47.4	0.1	48.6	0.1	49.4	0.1	0.8	0.1	0.6
0.3	33.3	35.8	37.4	38.0		5.6	5.1	4.8	4.7		38.2	0.3	42.3	0.3	45.7	0.3	47.6	0.3	2.1	0.4	1.5
(CTD) Cosine Distance, Triplet Loss, Defensive (R@1=97.4%)																					
0	50	50	50	50		1.5	1.5	1.5	1.5		50	50	50	50		0.5	0.5	0.5	0.5		
0.01	49.3	49.6	49.5	49.7		1.6	1.6	1.6	1.6		49.5	0.0	49.7	0.0	50.0	0.0	50.0	0.0	0.5	0.0	0.5
0.03	48.2	48.3	48.8	48.8		1.8	1.8	1.7	1.7		49.3	0.0	48.9	0.0	49.5	0.0	49.7	0.0	0.6	0.0	0.5
0.1	44.7	45.4	46.3	46.4		2.8	2.6	2.5	2.5		46.3	0.1	47.4	0.1	48.6	0.1	49.2	0.1	0.7	0.1	0.6
0.3	35.5	38.5	40.3	40.9		5.7	5.3	5.1	5.0		39.3	0.4	43.1	0.3	46.0	0.3	47.5	0.3	1.8	0.4	1.4

Table 17. The loss that directly suppresses embedding shift distance with MNIST dataset.

imum shift distance of embedding vectors, *i.e.*:

$$L_{\text{trip-es}} = L_{\text{triplet}}(q, c_p, c_n) + \sum_{x \in \{q, c_p, c_n\}} \left(\max_{r \in \Gamma} d(x+r, x) \right), \quad (22)$$

where the most severe distance shift incurred by adversarial perturbations is explicitly suppressed, in addition to a standard ranking loss term.

Experimental results (Tab. 17) show that cosine distance-based ranking models are more robust with this defense. However, we note that this loss may numerically explode on an Euclidean distance-based embedding model, as a strong adversary can gradually cause very large embedding shift distance. To mitigate such divergence issue, we also tried to add a balancing parameter to greatly scale down the second term of the loss function, but the instability problem was still not alleviated.

Due to the lack of universality, we leave this alternative defense in supplementary material as a pure discussion, and possible improvements as future work.