R^{2}TUA: Reconstruction-residual Based Targeted and Untargeted Attack Against Text-Image Person Re-Identification

Supplementary Material

In this supplementary material, we provide additional discussions on the following aspects:

- **Soft Clamp Function**: We detail the derivation of the soft clamp function and analyze a toy example to show how it works. In addition, we compare it with other clamp functions in experiments and discuss its limitations as well as areas for improvement.
- **Real-World Application Concerns**: We visualize attacked images under different perturbation thresholds ϵ and apply our attack to a real-world human-attributes detection system to demonstrate its practical attack potential.
- **Defence Analysis**: We apply R^{2}TUA as an adversarial-sample augmentation method to retrain a new RaSa [1] model, and evaluate its performance against R^{2}TUA attacks to demonstrate the effectiveness of adversarial sample augmentation.

1. Soft Clamp Function

In this section, we provide detailed analyses and additional experiments related to the proposed **S**oft **C**lamp **F**unction (SCF), which is designed for effective adversarial training. Specifically, we thoroughly explore the derivation and theoretical foundations of SCF, validate its practical effectiveness through comprehensive toy example comparisons and ablation studies, and address practical considerations and limitations.

1.1. Derivation of the Soft Clamp Function

1.1.1. Design objectives.

Assume that $x \in \mathbb{R}$ is one *un–clamped* perturbation element, and let $C(x)$ denote the clamp function. For adversarial training, we require an odd, continuous mapping. In addition, this mapping also needs to satisfy the following requirements:

1. Keep in–threshold signals unchanged, i.e., when $|x| < \epsilon$, $C(x) \to x \quad \text{and} \quad C'(x) \to 1 \quad ;$
2. Saturate outside the ℓ_∞ bound ϵ, i.e., when $|x| > \epsilon$, $C(x) \to \epsilon\,\mathrm{sgn}(x)$; $\lim_{|x|\to\infty} C(x) = \epsilon\,\mathrm{sgn}(x)$;
3. Avoid blocking gradients, i.e., $C'(x) \neq 0,\ \forall x \in \mathbb{R}$, so that back-propagation can still correct weights even when the forward value is already saturated.

1.1.2. Derivation Steps

Next, we discuss how to derive the clamp function from the above design objectives.

- **Step 1: Non-dimensionalization.** To simplify the derivation, we introduce a dimensionless variable $t = x/\epsilon$, transforming the clamping problem into finding a normalized odd mapping $c(t)$ that satisfies $c(0) = 0$, $c'(0) = 1$, and $\lim_{|t|\to\infty} c(t) = \mathrm{sgn}(t)$.
- **Step 2: Choosing the functional family.** To smoothly transition between the identity mapping within the boundary and saturation at infinity, we construct the normalized mapping $c(t)$ by combining linear identity behavior near zero and sign saturation at infinity, leading us to the following form $c(t) = t \cdot g(t^{2n})$. We require that $g(\cdot)$ be a positive even function, with $g(t^{2n}) \to 1$ when $t^{2n} < 1$ and $g(t^{2n}) \to 1/|t|$ when $t^{2n} > 1$. This yields:

$$g(t^{2n}) = \left(1 + |t|^{2n}\right)^{-\frac{1}{2n}}, \quad c(t) = \frac{t}{\sqrt[2n]{1 + t^{2n}}}, \quad n \in \mathbb{N}^+ \tag{1}$$

where n is a smoothness parameter (typically a large number). The mapping satisfies $c(0) = 0$ and $\lim_{|t|\to\infty} c(t) = \lim_{|t|\to\infty} \frac{t}{|t|(1+t^{-2n})^{\frac{1}{2n}}} = \mathrm{sgn}(t)$. Then, we can calculate its derivative:

$$\begin{aligned} c'(t) &= (1 + t^{2n})^{-\frac{1}{2n}} - \frac{t^{2n}}{(1 + t^{2n})^{1+\frac{1}{2n}}} \\ &= (1 + t^{2n})^{-\frac{2n+1}{2n}} \end{aligned} \tag{2}$$

Here $c'(0) = 1$. Since $c(t)$ is monotonically increasing, we have $c'(t) > 0$ globally. When $|t| \gg 1$, $c'(t) \sim |t|^{-(2n+1)}$ decays polynomially, while when $|t| < 1$, $c'(t) \approx 1$. This provides a smooth odd mapping that meets all the objectives. The parameter n controls the transition sharpness. Assuming $t \gg 1$, when $n = 1$, $c'(t) \sim |t|^{-3}$ and $c(t)$ behaves as a Sigmoid-like distortion activation function. When $n \to \infty$, $c'(t) \to 0$, and $c(t)$ approaches the conventional clamp function.
- **Step 3: Restoring physical units.** Resubstituting $t = x/\epsilon$ and scaling the output into $\pm\epsilon$ gives:

$$C(x) = \frac{\epsilon\, x}{\left(\epsilon^{2n} + x^{2n}\right)^{\frac{1}{2n}}}. \tag{3}$$

This is the SCF used in the paper. Its derivative is:

$$\frac{\partial C(x)}{\partial x} = \frac{\epsilon^{\,2n+1}}{\left(\epsilon^{2n} + x^{2n}\right)^{1+\frac{1}{2n}}} = \frac{C(x)}{x} - \frac{C(x)^{2n+1}}{\epsilon^{2n} \times x} \tag{4}$$

The function $C(x)$ quantitatively implements the differentiability and distortion-free properties required for robust optimization. These two formulations are consistent

Table 1. Forward output $f(x)$ and gradient $\partial L/\partial w$ ($y=0,\ w=1,\ b=0,\ \epsilon=1$).

Clamp Function	$f(x)$ (value)			$\partial L/\partial w$ (gradient)		
	$x=1$	$x=2$	$x=5$	$x=1$	$x=2$	$x=5$
Conventional clamp	1	1	1	0	0	0
Sigmoid	4.62×10^{-1}	$1-2.4\times10^{-1}$	$1-1.3\times10^{-2}$	3.9×10^{-1}	2.1×10^{-1}	1.3×10^{-2}
Tanh	7.62×10^{-1}	$1-3.6\times10^{-2}$	$1-9.1\times10^{-5}$	4.2×10^{-1}	7.1×10^{-2}	1.8×10^{-4}
SCF ($n=1$)	7.07×10^{-1}	$1-1.1\times10^{-1}$	$1-1.9\times10^{-2}$	3.5×10^{-1}	8.9×10^{-2}	7.5×10^{-3}
SCF ($n=10$)	9.66×10^{-1}	$1-4.8\times10^{-8}$	$1-2.0\times10^{-15}$	4.8×10^{-1}	4.8×10^{-7}	2.1×10^{-15}
SCF ($n=100$)	9.97×10^{-1}	$1-3.1\times10^{-63}$	$1-8.0\times10^{-143}$	5.0×10^{-1}	4.1×10^{-25}	2.4×10^{-57}

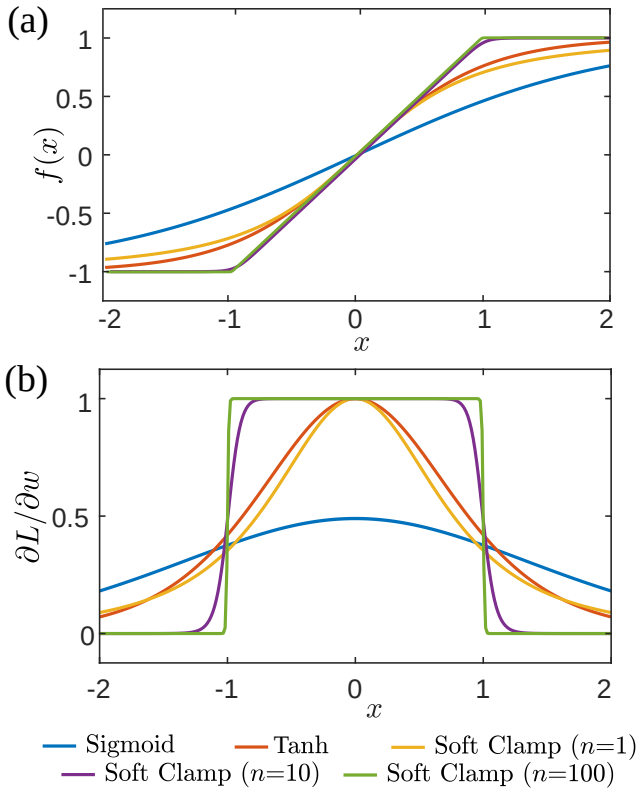

Figure 1. Outputs (a) and gradients (b) visualization of clamping functions for comparison.

with the implementation reported in the original R^{2}TUA work.

1.2. Toy Example Analysis

After introducing the proposed SCF, we now compare its gradient characteristics and forward process against other clamp functions (conventional clamp function [3], Sigmoid-scaling (Sigmoid) and Tanh-scaling (Tanh) [2]). This comparative analysis highlights the advantages of our proposed method.

Considering the exact gradient expression with mean squared error (MSE) losses, $L=\frac{(f(x)-y)^2}{2}$, $f(x)=\text{clamp}(wx+b)$, $\partial L/\partial w=(f(x)-y)f'(x)x$, with different

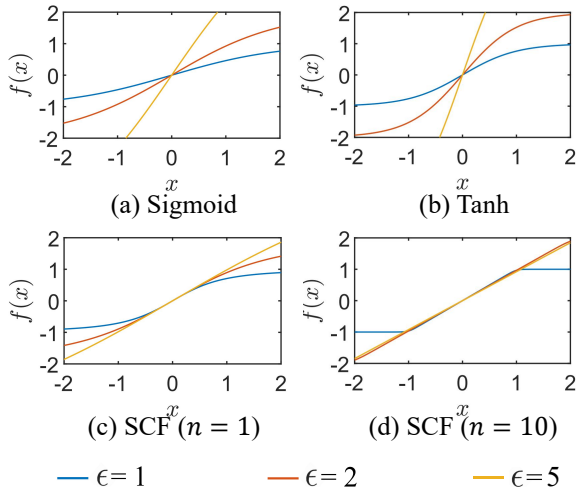

Figure 2. Outputs visualization of clamping functions under various ϵ.

clamp functions as clamp$(\cdot)$, we evaluate gradients at inputs $x=1, x=2$ and $x=5$ under the conditions $y=0$, $b=0$, and $w=1$. The numerical results are shown in Tab. 1, and the corresponding visualization is shown in Fig. 1. In addition, to examine the undistorted/distorted functionalities, we visualize the outputs under different ϵ values, as shown in Fig. 2.

As shown in Tab. 1, at $x=1$, the conventional clamp function immediately saturates the input to 1, resulting in zero gradient. The Sigmoid produces $f(x)\approx0.462$ with a gradient of 0.39, the Tanh function produces $f(x)\approx0.762$ with a gradient of 0.42, and the SCF with $n=1$ produces $f(x)\approx0.707$ with a gradient of 0.35. This behavior shows that, although the three functions compress their outputs within the threshold, there is a significant deviation between the compressed value and the input value. In addition, as shown in Fig. 2 (a-c), when ϵ changes, the compressed value will also change significantly, especially for Sigmoid and Tanh clamping functions, which prevents the application of a dynamic ϵ adjustment strategy in the process of training R^{2}TUA.

In contrast, when $n=10$ or 100, the SCF's output is closer to 1 ($f(x)\approx0.966$ for $n=10$ and 0.997 for

Table 2. Experiment results with different clamp functions.

Clamp Function	Untargeted				Targeted			
	R@1	R@5	R@10	Map	R@1	R@5	R@10	Map
Conventional Clamp	2.21	6.27	8.80	2.17	79.54	93.49	96.36	46.62
Sigmoid	22.63	34.11	39.52	15.38	0.20	0.39	1.30	0.40
Tanh	3.28	5.05	6.53	2.27	0.85	3.45	6.64	1.29
SCF (n=1)	0.47	2.13	3.78	0.88	84.97	96.45	97.66	52.23
SCF (n=10)	**0.11**	**1.14**	**2.16**	**0.60**	**89.43**	**97.72**	**98.47**	**54.89**
SCF (n=100)	0.32	1.33	2.32	0.63	83.73	95.90	97.59	52.34

$n = 100$), while the gradient drastically increases, approaching 0.48 for $n = 10$ and 0.5 for $n = 100$. As for the outputs with various ϵ, as shown in Fig. 2 (d), when $n = 10$, the SCF produces largely undistorted outputs across different ϵ values, even more so when $n = 100$. The above results show that SCF can maintain an almost unaltered output while preserving gradients.

This shows that SCF has two key advantages. First, it minimizes distortion and remains approximately an identity mapping within the $\pm\epsilon$ threshold, which makes adaptive ϵ adjustment during training possible. Second, and more importantly, it maintains non-zero gradients, which is critical for efficient optimization. In contrast, functions like Sigmoid and Tanh introduce severe distortion, blocking the application of dynamic ϵ adjustment, while the conventional clamp function cuts off gradients and hinders optimization.

At $x = 2$, the SCF with $n = 1$ saturates with $f(x) \approx 1 - 1.1 \times 10^{-1}$ and retains a gradient of 0.09, which lie between those of Sigmoid ($f(x) \approx 1 - 2.4 \times 10^{-1}$ and gradient 0.21) and Tanh ($f(x) \approx 1 - 3.6 \times 10^{-2}$ and gradient 0.07). When $n = 10$, the SCF produces an output very close to 1 ($1 - 4.8 \times 10^{-8}$), with a small gradient of 4.8×10^{-7}, which further demonstrates its gradient-preserving ability while maintaining saturation. As for $n = 100$, under the constraints of the effective digit numbers, the SCF ($f(x) \approx 1 - 3.1 \times 10^{-63}$ and gradient 4.1×10^{-25}) is effectively the same as the conventional clamp, saturating the output to 1 and resulting in a gradient that is technically zero.

At the extreme input value $x = 5$, Sigmoid and Tanh functions approach saturation. The Sigmoid function outputs $f(x) \approx 1 - 1.3 \times 10^{-2}$ with a gradient of 0.013. The Tanh function outputs $f(x) \approx 1 - 1.9 \times 10^{-5}$ with a gradient of 1.8×10^{-4}. In contrast, the SCF with $n = 10$ achieves near-perfect saturation: $f(x) \approx 1 - 2.0 \times 10^{-15}$, and its gradient becomes negligible (2.1×10^{-15}). However, such extreme values are very rare in practice. Typically, perturbations generated by an unoptimized attacker are normally distributed. Moreover, when extreme perturbations do occur, they sharply increase the Structural Similarity Index Measure (SSIM) loss during optimization. This increase penalizes the attacker, discouraging the generation of such extreme perturbations.

Overall, for $n = 10$, the SCF function achieves a balance between keeping outputs undistorted and preserving gradients. This confirms that the method is essential for gradient-based adaptive ϵ adjustment learning algorithms.

1.3. Comparison Experiments

We conduct experiments on R^{2}TUA against the RaSa model [1] on the CUHK-PEDES dataset [4] with different clamp functions, while keeping all other experimental settings the same as those proposed in the paper. The experiment results are shown in Tab. 2.

From Tab. 2, we can see the performance differences among the evaluated clamp functions. Notably, both Sigmoid and Tanh functions fail to adapt directly to our proposed dynamic ϵ adjustment strategy, leading to convergence issues in R^{2}TUA training. Specifically, the Sigmoid function performs particularly poorly and is unable to achieve effective untargeted attacks.

In contrast, the conventional clamp function basically achieves the requirements of targeted and untargeted attacks, and its performance is significantly improved compared with that of similar activation functions (Sigmoid and Tanh), especially for targeted attacks, where the performance is close to 80%. However, our proposed SCF performs much better, demonstrating excellent adaptability and significantly improved performance across all metrics. Among the SCF variants, the choice of smoothness parameter $n = 10$ consistently outperforms both $n = 1$ and $n = 100$.

More specifically, when $n = 1$, the SCF exhibits relatively weaker performance on untargeted attacks but stronger targeted attack performance. Conversely, at $n = 100$, the situation is reversed, showing slightly stronger untargeted performance but weaker targeted results. However, these differences are minor and do not indicate a clear preference for either targeted or untargeted attack scenarios. Overall, the experimental results emphasize the effectiveness of SCF and confirm that selecting $n = 10$ as the smoothness-controlling parameter is both practical and beneficial for balancing performance across different attack settings.

1.4. Limitations and Discussions

The above analysis demonstrates the effectiveness of the proposed SCF. However, there are still some limitations that we need to discuss.

First, the power exponent that controls smoothness in the SCF, i.e., n, can lead to computational overflow. Specifically, when calculating the denominator $\left(\epsilon^{2n} + x^{2n}\right)^{\frac{1}{2n}}$, we need to compute both ϵ^{2n} and x^{2n}. Clearly, for large n (e.g., $n = 10$), if ϵ or $|x|$ is very large, $\epsilon^{2n} + x^{2n}$ can cause overflow during computation.

To mitigate the risk of such precision overflow, we propose a relaxation approach by scaling both the numerator and denominator by ϵ. This is done by introducing a dimensionless variable $t = x/\epsilon$, transforming the calculation of $C(x)$ into the calculation of $c(t)$ in Eq. 1, which is easier to compute and avoids extreme values. Specifically, the SCF is transformed into $\epsilon \times c(t) = \frac{\epsilon t}{\sqrt[2n]{1+t^{2n}}}$, ensuring that we avoid excessively large $|x|$ or large ϵ. As to the scenario of large $|x|$ with small ϵ, from an engineering perspective, we observed that, during the SCF application in our work, as long as the initial value of ϵ is sufficiently large (e.g., 0.1), and ϵ decays smoothly during training (with each decay rate < 0.5), we did not observe numerical overflow under these settings in our experiments. This is because the normal distribution of weight initialization in the reconstructor causes the generated perturbations to be approximately normally distributed, with variance much less than 1. Therefore, the probability of obtaining large values (e.g., $x > 5 \times 0.1$) in the perturbation distribution is negligible, which in turn leads to a low probability of overflow due to large x. In addition, constraining the structural similarity metric loss output by the reconstructor naturally reduces the variance of the perturbations during training, ensuring that the probability of obtaining large values is reduced during training.

Nevertheless, to fully eliminate the possibility of numerical overflow, we propose an additional safeguard: embedding a conventional clamp function with a threshold of 5ϵ before the SCF. This conventional clamping ensures that any values that would cause overflow are clipped, thus preventing overflow. Importantly, this modification does not undermine the functionality of the SCF, as gradients when $|x| > 5\epsilon$ are smaller than 2×10^{-15}, which can be considered practically equivalent to zero in engineering applications. Therefore, by incorporating this conventional clamp, we avoid overflow issues while retaining the SCF's key feature of mitigating large gradient variations (LGV), which is essential for stable training.

In summary, the proposed SCF is effective under various conditions, but for extreme edge cases, the careful adjustment of ϵ and the inclusion of a conventional clamp provide a robust solution to computational challenges, ensuring that the function remains numerically stable and suitable for practical use in adversarial training.

Figure 3. Visualization of attacked images, where $\epsilon = 0$ denotes no attack.

2. Real-World Application Concerns

To complement the theoretical analysis and experiments on academic datasets in the main paper, we further examine two practical aspects that are critical for real-world deployment. First, we visualize perturbed images under different ℓ_∞ thresholds to assess whether the attack remains inconspicuous to human observers. Second, we probe an online human-attributes detection system (from Baidu company) to evaluate transferability to a commercial-grade pipeline and to quantify qualitative failure modes that may arise in practice.

These studies aim to answer two questions: whether the perturbations produced by R^{2}TUA can be deployed without being noticed during normal usage, and whether the attack maintains efficacy when the target system is deployed in the real world. The results demonstrate that the perturbations are visually inconspicuous at moderate ϵ and that the attack induces systematic attribute errors on a black-box service, indicating practical risk beyond laboratory settings.

2.1. Attack Visualization

We visualize representative examples under $\epsilon \in \{0, 8/255, 16/255\}$ to study perceptual impact (Fig. 3). The unattacked images ($\epsilon = 0$) serve as references. At $\epsilon = 8/255$, the images remain visually indistinguishable from the unattacked images at normal scale. Edges, global color balance, and salient object boundaries are preserved. This behavior indicates that R^{2}TUA can be deployed in realistic scenarios without arousing human suspicion.

When ϵ increases to $16/255$, slight high-frequency deviations may be noticed under zoom-in comparison, particularly around fine textures and smooth regions. However, when viewed in isolation without the original for direct comparison, these changes are difficult to detect. Taken

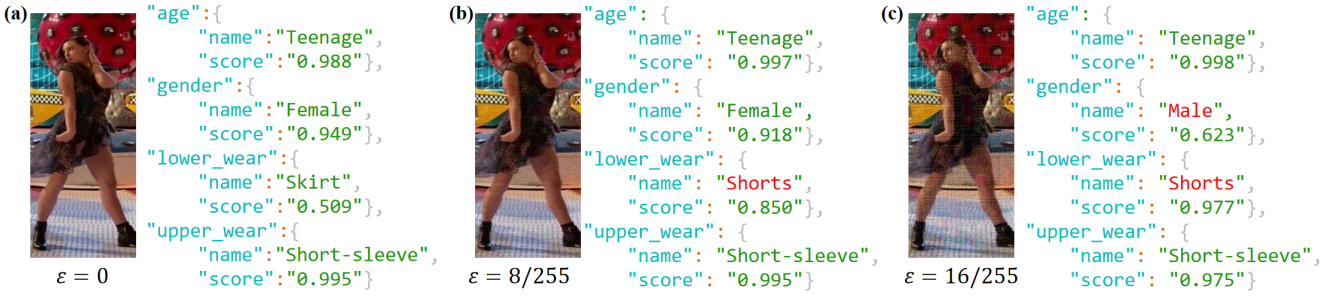

Figure 4. Attacks on a human-attributes detection system, where dict values are attributes detection results. R^{2}TUA is trained on RaSa [1].

together, these observations show a favorable trade-off: perturbations that are perceptually mild yet sufficiently structured to induce model errors.

2.2. Real World System Attack

We evaluate transferability on an online human-attributes detection system by submitting the original image and perturbed images produced with $\epsilon = 8/255$ and $\epsilon = 16/255$; see Fig. 4 for results. The service returns a dictionary of predicted attributes, and we choose 4 main attributes, namely age, gender, upper wear and lower wear. For the adversarial prompt of R^{2}TUA: "A man with yellow coat and a pair of blue shorts is walking across the street," the original image yields correct attributes. Under $\epsilon = 8/255$, the system begins to produce inconsistent predictions for lower-body clothing. At $\epsilon = 16/255$, the mispredictions become more pronounced and propagate to higher-level attributes, including the flip of the predicted gender. The progression with increasing ϵ reflects the attack's mechanism: R^{2}TUA preserves global structure and overall appearance while injecting targeted, fine-scale features that mislead the commercial attribute prediction system. The outcome confirms that the attack is not confined to a specific research model and can induce consequential semantic errors in a black-box, real-world system.

3. Defence Analysis

To further broaden the applicability of R^{2}TUA and to prevent uncontrolled behavior in real deployments, we provide a pragmatic, deployable defence option: perform standard adversarial training with attack-informed data augmentation derived from R^{2}TUA, without altering the model architecture or system interfaces. This procedure is not positioned as a core scientific contribution; rather, it offers a default hardening path for practical systems and reveals a new area that we can explore.

As shown in Tab. 3, our defence scheme preserves parity with random-noise augmentation (R@1 75.06 vs. 75.82). For *untargeted* attacks, after our defence, these metrics increase substantially (R@1 54.21 vs. 21.82), evidencing reduced attack success and improved robustness. For *targeted* attacks, after our defence, these metrics decrease markedly (R@1 1.30 vs. 24.76 with consistent drops in R@5/R@10/mAP), showing that targeted objectives are harder to achieve against the defended model.

Table 3. Model performance on CUHK [4] with adversarial training, where random noises follow the normal distribution with a mean of 0 and a standard deviation of 8/255. The R^{2}TUA is trained on APTM [5] model.

Augmentation	Category	R@1	R@5	R@10	mAP
RandomNoise	None*	75.82	90.29	93.50	68.27
	Untargeted	21.82	39.23	48.07	16.67
	Targeted	24.76	50.13	61.32	14.19
R^{2}TUA	None*	75.06	89.75	93.57	67.86
	Untargeted	54.21	74.64	81.60	44.77
	Targeted	1.30	6.25	11.94	1.54

None* denotes the performances of without attack.

References

[1] Yang Bai, Min Cao, Daming Gao, Ziqiang Cao, Chen Chen, Zhenfeng Fan, Liqiang Nie, and Min Zhang. Rasa: relation and sensitivity aware representation learning for text-based person search. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, pages 555–563, 2023. 1, 3, 5

[2] 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] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 2

[4] Shuang Li, Tong Xiao, Hongsheng Li, Bolei Zhou, Dayu Yue, and Xiaogang Wang. Person search with natural language description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1970–1979, 2017. 3, 5

[5] Shuyu Yang, Yinan Zhou, Zhedong Zheng, Yaxiong Wang, Li Zhu, and Yujiao Wu. Towards unified text-based person retrieval: A large-scale multi-attribute and language search benchmark. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 4492–4501, 2023. 5