Cross-Modality Attack Boosted by Gradient-Evolutionary Multiform Optimization

Anonymous submission

Supplementary Materials

Algorithm 1: Multiform Attack Optimization Framework

Input: $\mathcal{D}_{\text{train}}$: Training dataset for gradient optimization, $\mathcal{D}_{\text{evo}}$: Training dataset for evolutionary search, $\mathcal{M}_{\text{grad}}$: Model for gradient optimization, $\mathcal{M}_{\text{evo}}$: List of models for evolutionary search, ϵ: Perturbation limit, α: Learning rate, β: Momentum decay coefficient, max_iter: Maximum number of iterations, k: Maximum number of perturbed pixels per iteration, $\mathcal{P}$: Initial population size, G: Number of generations in evolutionary search
Output: δ: Optimized perturbation after all iterations
Initialization: $\delta \leftarrow Rand(0,1)$, $v \leftarrow 0$ ▷ Exponential moving average of the gradient
for $t = 0$ **to** max_iter $- 1$ **do**
    Fetch batch x from $\mathcal{D}_{\text{train}}$
    Compute the loss $\mathcal{L}_{\text{meta}}(\delta, x)$ using $\mathcal{M}_{\text{grad}}$
    $v_{t+1} \leftarrow \beta v_t + (1-\beta)\frac{\nabla_\delta \mathcal{L}_{\text{meta}}}{\|\nabla_\delta \mathcal{L}_{\text{meta}}\|_1}$
    $\delta_{t+1} \leftarrow \text{clip}(\delta_t + \alpha \cdot \text{sign}(v_{t+1}), -\epsilon, \epsilon)$
    if $t \mod iter == 0$ **then** ▷ Execute evolutionary search every $iter$ iterations
        Fetch batch x_{evo} from $\mathcal{D}_{\text{evo}}$
        Initialize population $\mathcal{P}$
        Generate new individuals through crossover and mutation
        for each individual $\eta \in \mathcal{P}$ **do**
            Evaluate η using all models in $\mathcal{M}_{\text{evo}}$ on x_{evo}
        end for
        Perform non-dominated sorting to select the best η
        $\delta_t \leftarrow \delta_t + \eta$ ▷ Update δ_t with the best η found
    end if
end for
Output the optimized perturbation δ

Description of Experimental Equipment

Our device is equipped with three NVIDIA A40 GPUs, each with 48GB of memory.

Details of the Evolutionary Algorithm

Initialization

The attack method initializes by setting the number of perturbed pixels to k and constructing a set $\mathcal{P}$ of s solutions. Each solution is created by uniformly sampling pixel locations from the set $\{1, \cdots, h \cdot w\}$. The initial perturbation values for each color channel are randomly sampled from the set $\{-1, 0, 1\}$:

$$\mathcal{P} = \{\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_s\}, \quad \mathcal{P}_i \in \{-1, 0, 1\} \tag{1}$$

where:

- $\mathcal{P}$: The set of initial solutions.
- s: The total number of solutions.

When invoking the evolutionary process, we set a fixed number of iterations to 200.

Crossover

The crossover operation aims to generate new solutions by combining traits from two parent solutions. This process helps in exploring the solution space and inheriting beneficial traits from the parents.

Crossover combines traits from two parent solutions P_a and P_b to generate new solutions:

$$M' = (M_a \cup M_b) \setminus (M_a \cap M_b) \tag{2}$$

$$\Delta' = \Delta_a \cup \Delta_b \tag{3}$$

where:

- M_a, M_b: Sets of pixel locations from the parent solutions P_a and P_b, respectively.
- M': The set of pixel locations for the new solution.
- Δ_a, Δ_b: Sets of perturbation values corresponding to M_a and M_b.
- Δ': The set of perturbation values for the new solution.

Mutation

The mutation operation introduces variations by modifying a subset of pixel locations. This process helps in exploring new regions of the solution space and maintaining diversity among solutions.

Mutation introduces variations to solutions by modifying a subset of pixel locations:

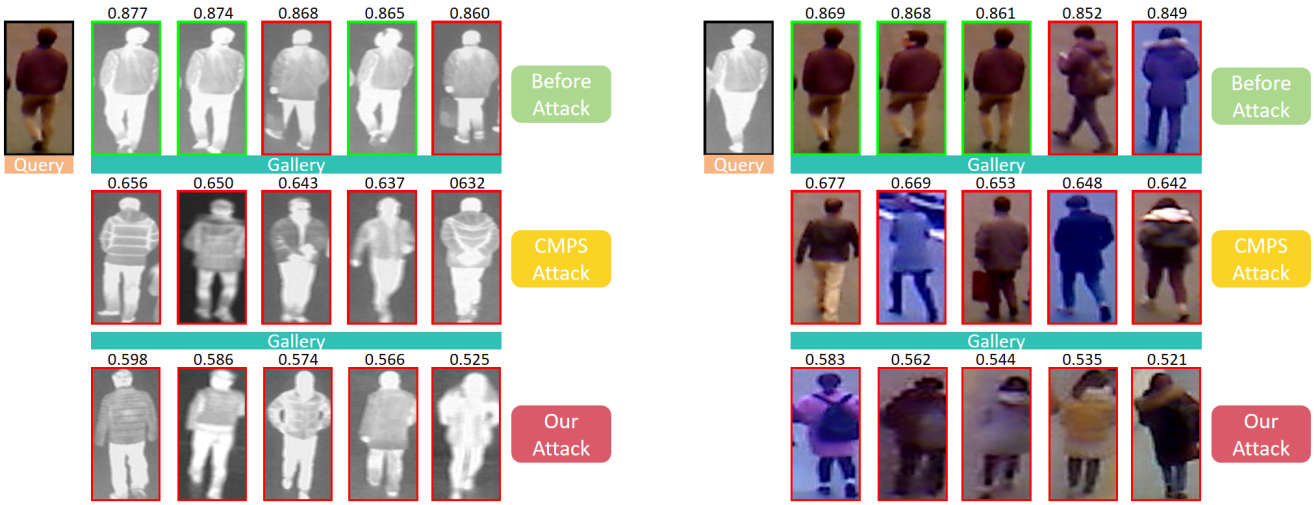

Figure 1: Comparison with the state-of-the-art attack method CMPS attack retrieval results. Before and after our attack on RegDB, the top five predictions of DDAG (a state-of-the-art cross-modality ReID baseline). Green boxes indicate correctly matched images, while red boxes indicate mismatched images. The left image corresponds to visible modality retrieving non-visible modality, and the right image vice versa.

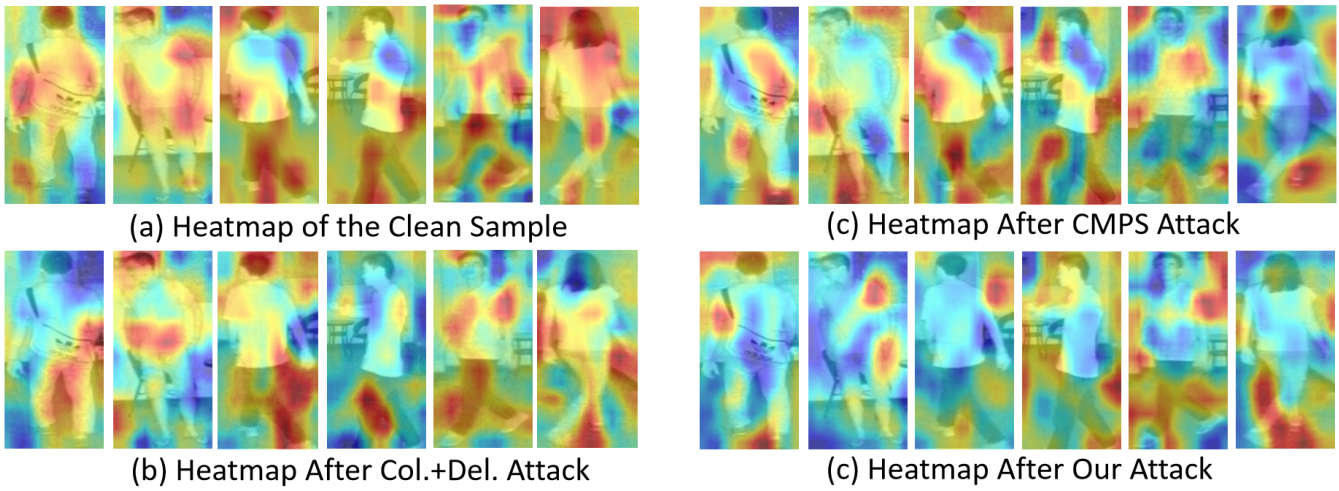

Figure 2: Comparison with state-of-the-art adversarial attack methods on attention heatmaps.

Table 1: Results for attacking cross-modality ReID systems on the SYSU dataset. It reports on visible images querying infrared images and vice versa. Rank at r accuracy (%) and mAP (%) are reported. The perturbation ϵ is set to 8.

Settings		Visible to Infrared				Infrared to Visible			
Method	Venue	$r=1$	$r=10$	$r=20$	mAP	$r=1$	$r=10$	$r=20$	mAP
AGW baseline (Ye et al. 2022)	TPAMI 2022	47.50	84.39	92.14	47.65	54.17	91.14	95.98	62.97
M-FGSM attack (Bai et al. 2020)	TPAMI 2020	25.79	49.04	57.96	19.24	20.56	38.91	46.35	15.89
LTA attack (Gong, Huang, and Chen 2022)	CVPR 2022	8.42	21.25	27.98	9.16	20.92	32.18	36.80	15.24
ODFA attack (Zheng et al. 2023)	IJCV 2023	25.43	47.49	56.38	19.00	14.62	29.92	36.42	11.35
Col.+Del. attack (Yang et al. 2023)	TPAMI 2023	3.23	14.48	20.15	3.27	4.12	16.85	21.27	3.89
CMPS attack (Gong et al. 2024)	Arxiv 2024	1.11	8.67	16.14	1.41	1.31	7.47	10.36	1.23
Our attack*	————	1.10	7.42	14.46	1.27	1.25	6.34	9.39	1.12
Our attack	————	1.02	7.24	14.28	1.13	1.18	6.02	9.17	1.11
DDAG baseline (Ye et al. 2020)	ECCV 2020	54.75	90.39	95.81	53.02	61.02	94.06	98.41	67.98
M-FGSM attack (Bai et al. 2020)	TPAMI 2020	28.36	52.47	60.76	23.11	24.85	40.74	49.23	18.40
LTA attack (Gong, Huang, and Chen 2022)	CVPR 2022	10.54	23.08	30.47	12.28	18.93	34.12	41.52	15.24
ODFA attack (Zheng et al. 2023)	IJCV 2023	27.75	50.26	59.14	22.30	17.62	32.64	40.03	14.83
Col.+Del. attack (Yang et al. 2023)	TPAMI 2023	4.28	16.12	21.36	3.97	6.28	19.53	25.61	5.21
CMPS attack (Gong et al. 2024)	Arxiv 2024	1.62	7.59	14.46	1.84	1.45	7.71	10.72	1.25
Our attack*	————	1.54	6.72	12.66	1.61	1.40	7.68	10.43	1.18
Our attack	————	1.23	6.38	12.06	1.22	1.31	7.44	10.15	1.06

$$M'' = (M' \setminus A) \cup B, \quad \Delta'' = (\Delta' \setminus \Delta_A) \cup \Delta_B \tag{4}$$

where:

- M': The set of pixel locations before mutation.
- A: A randomly selected subset of pixel locations to be replaced.
- B: A new subset of pixel locations to be introduced (obtained through random selection).
- M'': The set of pixel locations after mutation.
- Δ': The set of perturbation values before mutation.
- Δ_A: The perturbation values corresponding to A.
- Δ_B: The perturbation values corresponding to B.
- Δ'': The set of perturbation values after mutation.

Evaluation

The evaluation process calculates the objective vector $\Phi(f_{adv}) = (\tilde{\mathcal{D}}(f_{adv}), \tilde{\mathcal{S}}(f_{adv}), \|\eta\|_2, \|\eta\|_0)^T$ for each solution to measure its quality. This step is crucial for selecting the best solutions.

Selection

Algorithm 2: Non-Dominating Sorting for Multiform Attack

Input: Combined population $\mathcal{P}$, objective vectors Φ
Output: Set of fronts $\mathcal{S}_f$
$\mathcal{S}_f \leftarrow \{\}$ // Initialize the set of fronts
for $p \in \mathcal{P}$ **do**
  $S_p \leftarrow \{\}$ // Set of p dominated solutions
  $n_p \leftarrow 0$ // Domination counter of p
  for $q \in \mathcal{P}$ **do**
    if DOMINATES(p, q, Φ) **then** //if p dominates q
      $S_p \leftarrow S_p \cup \{q\}$
    else if DOMINATES(q, p, Φ) **then**
      $n_p \leftarrow n_p + 1$
    end if
  end for
  if $n_p == 0$ **then**
    $p_{\text{rank}} \leftarrow 1$ // p belongs to the first front
    $\mathcal{S}_{f1} \leftarrow \mathcal{S}_{f1} \cup \{p\}$
  end if
end for
$i \leftarrow 1$ // Initialize front counter
while $\mathcal{S}_{fi} \neq \emptyset$ **do**
  $Q \leftarrow \{\}$ // Store solutions of the next front
  for $p \in \mathcal{S}_{fi}$ **do**
    for $q \in S_p$ **do**
      $n_q \leftarrow n_q - 1$
      if $n_q == 0$ **then**
        $q_{\text{rank}} \leftarrow i + 1$
        $Q \leftarrow Q \cup \{q\}$
      end if
    end for
  end for
  $i \leftarrow i + 1$
  $\mathcal{S}_{f(i+1)} \leftarrow Q$
end while
return $\mathcal{S}_f$

Algorithm 3: Function to Determine if One Solution Dominates Another

```
function DOMINATES(x, y, Φ)
    isBetter ← False
    for i ← 1 to length(Φ) do
        if Φ_i(x) > Φ_i(y) then
            isBetter ← True
        else if Φ_i(x) < Φ_i(y) then
            return False
        end if
    end for
    return isBetter
end function
```

The selection process evaluates solutions and chooses the best individuals to pass their genetic material to the next generation. Initially, non-dominated sorting is used to select the best solutions from the combined parent and offspring populations.

Selection uses non-dominated sorting to choose the best solutions for the next generation:

$$P' = \text{NonDominatedSort}(P \cup O) \tag{5}$$

where:

- P and O are the parent and offspring populations, respectively.
- P': The new population of solutions after selection.
- NonDominatedSort: A sorting method that selects non-dominated solutions based on their objective vectors.

The selection process can be represented as:

$$\text{Selected Solutions}\eta = \arg\min_{s \in P'} \Phi(\mathcal{F}(x + \delta + \eta_s)) \tag{6}$$

where:

- P' is the set of non-dominated solutions selected by non-dominated sorting.

In this way, we first use non-dominated sorting to select a set of non-dominated solutions and then choose the solutions with the smallest objective function values from these non-dominated solutions.

Feasibility Analysis of Evolutionary Search

Here, we simplify the problem and appropriately reformulate it to facilitate analysis.

Objective of Sparse Perturbations

The goal of introducing sparse perturbations is to minimize the number of changes made to the input while maximizing the error induced across various models. The fitness function, which evaluates the effectiveness of a perturbation across multiple models, is central to this process.

Fitness Function

Let Φ represent a collection of models $\{M_1, M_2, \ldots, M_k\}$, where each model M_i has an associated misclassification rate $r_i(x)$ for an input x. The fitness function $f(x)$ for a perturbation vector x is defined as:

$$f(x) = \sum_{i=1}^{k} w_i \cdot r_i(x) - \lambda \cdot \|x\|_0 \tag{7}$$

Here, $\|x\|_0$ denotes the sparsity of the perturbation vector (the number of non-zero elements), w_i is the weight associated with model M_i, reflecting its importance in the overall fitness, and λ is a regularization parameter that controls the significance of sparsity.

Generation of Sparse Perturbations

Evolutionary algorithms utilize selection, crossover, and mutation operations to generate new perturbations. The mutation operation, designed to maintain sparsity while enhancing fitness, is defined as:

$$x' = x + \delta \tag{8}$$

where δ is a small change vector, typically non-zero in only a few components of x. δ is chosen to maximize $f(x + \delta)$.

Convergence Analysis

By appropriately choosing the parameters λ and w_i, evolutionary search can converge to high-quality sparse perturbations. Theoretically, consider the change in the fitness function over iterations. assuming:

$$\lim_{t \to \infty} f(x^{(t)}) = f^* \tag{9}$$

where $x^{(t)}$ is the perturbation vector after the t-th iteration, and f^* represents the optimal achievable fitness.

To ensure that each iteration does not decrease the fitness, we need:

$$f(x^{(t+1)}) \geq f(x^{(t)}) \tag{10}$$

This indicates that the fitness is non-decreasing over iterations, suggesting that the algorithm is at least locally optimal. By designing δ to ensure that $f(x+\delta) > f(x)$, this can be achieved. This can be specifically implemented through techniques such as non-dominated sorting to select the most effective perturbations. Please note that our goal is to fine-tune the universal perturbation using the search process; in fact, we do not need to find the perturbation with the optimal fitness value.

Through the analysis provided, we theoretically demonstrate that evolutionary search can generate effective sparse perturbations that are effective across multiple models.

Feasibility Analysis of Enhancing Attack Transferability

Here, we simplify the problem and appropriately reformulate it to facilitate analysis.

Objective Definition

The goal is to find a fine-tuned perturbation δ_f that significantly increases the misclassification rate across multiple models while maintaining the sparsity of the perturbation. For this purpose, we introduce a fitness function:

$$f(\delta_f) = \sum_{i=1}^{k} w_i \cdot r_i(\delta_u + \delta_f) - \lambda \cdot \|\delta_f\|_0 \tag{11}$$

where $\|x\|_0$ denotes the sparsity of the perturbation vector (the number of non-zero elements), w_i is the weight associated with model M_i, reflecting its importance in the overall fitness, and λ is a regularization parameter that controls the significance of sparsity.

Complementarity Measure α_i

To quantify the complementarity effect of the fine-tuned perturbation δ_f on the universal perturbation δ_u, we define the complementarity measure α_i as:

$$\alpha_i = \frac{r_i(\delta_u + \delta_f) - r_i(\delta_u)}{r_i(\delta_u)} \tag{12}$$

Here, α_i represents the increase in the misclassification rate of model M_i when the fine-tuned perturbation δ_f is added to the universal perturbation δ_u. Ideally, α_i should be significantly greater than 0, indicating a strong complementarity effect.

Optimization of the Fitness Function

The goal of the evolutionary search is to maximize the fitness function $f(\delta_f)$. By selecting appropriate mutation vectors δ and performing crossover operations, we ensure that each iteration finds a better fine-tuned perturbation δ_f. Specifically, we aim to find δ_f such that:

$$\sum_{i=1}^{k} w_i \cdot r_i(\delta_u + \delta_f) - \lambda \cdot \|\delta_f\|_0 \tag{13}$$

is maximized.

Theoretical Guarantee

Through selection, crossover, and mutation operations, evolutionary search can effectively optimize the perturbation vector δ_f to exhibit a high complementarity measure α_i across multiple models.

Specifically, for each iteration, we have:

$$\begin{aligned} \alpha_i^{(t+1)} &= \frac{r_i(\delta_u + \delta_f^{(t+1)}) - r_i(\delta_u)}{r_i(\delta_u)} \\ &\geq \alpha_i^{(t)} = \frac{r_i(\delta_u + \delta_f^{(t)}) - r_i(\delta_u)}{r_i(\delta_u)} \end{aligned} \tag{14}$$

This indicates that the complementarity measure of δ_f increases with each iteration, thereby significantly improving the misclassification rate across multiple models.

By appropriately choosing the parameters λ and w_i, evolutionary search can converge to high-quality sparse perturbations. Theoretically, considering the change in the fitness function over iterations, we assume:

$$\lim_{t \to \infty} f(x^{(t)}) = f^* \tag{15}$$

where $x^{(t)}$ is the perturbation vector after the t-th iteration, and f^* represents the optimal achievable fitness.

To ensure that each iteration does not decrease the fitness, we need:

$$f(x^{(t+1)}) \geq f(x^{(t)}) \tag{16}$$

This indicates that the fitness is non-decreasing over iterations, suggesting that the algorithm is at least locally optimal. By designing δ to ensure $f(x + \delta) > f(x)$, this can be achieved. Specifically, this can be implemented through non-dominated sorting to select the most effective perturbations.

By selecting appropriate mutation vectors δ and performing crossover operations, evolutionary search can effectively optimize the perturbation vector δ_f to exhibit a high complementarity measure α_i across multiple models, i.e.:

$$\alpha_i = \frac{r_i(\delta_u + \delta_f) - r_i(\delta_u)}{r_i(\delta_u)} \gg 0 \tag{17}$$

This indicates that the fine-tuned perturbation significantly increases the misclassification rate for each model, thereby improving the overall attack effectiveness.

Through the above mathematical analysis and theoretical derivation, we demonstrate that evolutionary search can find fine-tuned perturbations δ_f with a high complementarity measure α_i, effectively refining the universal perturbation δ_u, thus achieving better transferability and attack effectiveness across multiple models. This method leverages the global search capability of evolutionary algorithms and the complementarity between different models, ensuring that the perturbations are effective across various models.

CnMix Processing Algorithm

Here we provide the detailed processing workflow of Cnmix, which generates new images by randomly mixing the three channels of the images. For specific details, refer to Alg. 4.

Discussion

Ethical Considerations

In this study, we introduce a novel cross-modal adversarial attack method that enhances the transferability and concealment of adversarial attacks through a gradient-evolutionary multiform optimization framework. This research offers a new perspective on understanding and enhancing the security of cross-modal systems, but it also raises a series of ethical and safety concerns about the potential negative impacts of adversarial attack techniques. Adversarial attack technology can be maliciously exploited, posing a serious threat to public safety.

However, we recognize the positive value of adversarial attack research. It reveals vulnerabilities in existing systems,

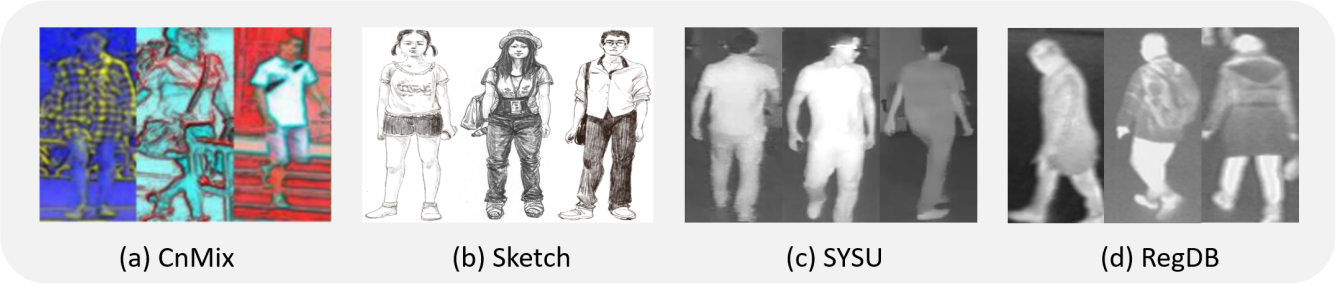

Figure 3: Examples of the four different modalities corresponding to the four datasets used in this paper.

prompting academia and industry to make in-depth improvements to the robustness of machine learning models. The positive impact of this study lies in its potential to combine adversarial training with the attack methods presented to enhance system security and bring positive social impacts. Therefore, we emphasize the importance of conducting adversarial attack research within an ethical framework and encourage further development of defensive technologies to build a safer and more reliable technological environment.

Limitations and Future Work

Here, we need to acknowledge the limitations of the proposed method and identify potential areas for future research. Firstly, current attack techniques mainly focus on enhancing the stealthiness of perturbations for RGB images. In other types of images, such as infrared or thermal images in monochromatic modes, the invisibility of perturbations remains a challenge. This is because these modalities typically lack rich color information, making even subtle perturbations noticeable. Future research needs to explore how to effectively hide adversarial perturbations in these different image modalities to improve the applicability and stealthiness of attacks.

Secondly, as the number of modalities processed increases, the computational demand rises sharply, which can lead to significant resource consumption in practical applications. The high demand for computational resources in current methods limits their feasibility in resource-constrained environments. Therefore, future work could explore more efficient algorithms by incorporating state-of-the-art evolutionary computation methods to reduce the computational burden when handling large-scale or multi-modal data.

References

Bai, S.; Li, Y.; Zhou, Y.; Li, Q.; and Torr, P. H. 2020. Adversarial metric attack and defense for person re-identification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(6): 2119–2126.

Gong, Y.; Huang, L.; and Chen, L. 2022. Person re-identification method based on color attack and joint defence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4313–4322.

Gong, Y.; et al. 2024. Cross-Modality Perturbation Synergy Attack for Person Re-identification. *arXiv preprint arXiv:2401.10090*.

Yang, F.; Weng, J.; Zhong, Z.; Liu, H.; Wang, Z.; Luo, Z.; Cao, D.; Li, S.; Satoh, S.; and Sebe, N. 2023. Towards robust person re-identification by defending against universal attackers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4): 5218–5235.

Ye, M.; Shen, J.; Crandall, D. J.; Shao, L.; and Luo, J. 2020. Dynamic Dual-Attentive Aggregation Learning for Visible-Infrared Person Re-Identification. In *European Conference on Computer Vision (ECCV)*.

Ye, M.; Shen, J.; Lin, G.; Xiang, T.; Shao, L.; and Hoi, S. C. 2022. Deep learning for person re-identification: A survey and outlook. *IEEE transactions on pattern analysis and machine intelligence*, 44(6): 2872–2893.

Zheng, Z.; Zheng, L.; Yang, Y.; and Wu, F. 2023. U-Turn: Crafting Adversarial Queries with Opposite-Direction Features. *International Journal of Computer Vision*, 131(4): 835–854.

Algorithm 4: Random Channel Mixing of Grayscale and Sketch Images

Input: img (Input RGB image), G (Probability of converting to grayscale), G_{rgb} (Probability of mixing grayscale and RGB), S_{rgb} (Probability of mixing sketch and RGB)
Output: $output\_img$ (Transformed image)
function TOSKETCH(img)
    $img\_np \leftarrow$ Convert img to numpy array
    $img\_inv \leftarrow 255 - img\_np$ ▷ Invert image colors
    $img\_blur \leftarrow$ Gaussian blur of img_inv
    $img\_blend \leftarrow \frac{img\_np}{255 - img\_blur} \times 256$ ▷ Blend original and blurred images
    return Convert img_blend to image
end function
function RANDOM_CHOOSE($r, g, b, gray\_or\_sketch$)
    $p \leftarrow [r, g, b, gray\_or\_sketch, gray\_or\_sketch]$
    $idx \leftarrow [0, 1, 2, 3, 4]$
    Shuffle idx
    return Merge channels $p[idx[0]], p[idx[1]], p[idx[2]]$ into RGB image
end function
function FUSE_RGB_GRAY_SKETCH(img, G, G_{rgb}, S_{rgb})
    $r, g, b \leftarrow$ Split img into RGB channels
    $gray \leftarrow$ Convert img to grayscale
    $p \leftarrow$ Random value between 0 and 1
    if $p < G$ **then**
        return Merge $gray, gray, gray$ into RGB image
    else if $p < G + G_{rgb}$ **then**
        $output\_img \leftarrow$ RANDOM_CHOOSE($r, g, b, gray$)
        return $output\_img$
    else if $p < G + G_{rgb} + S_{rgb}$ **then**
        $sketch \leftarrow$ TOSKETCH($gray$)
        $output\_img \leftarrow$ RANDOM_CHOOSE($r, g, b, sketch$)
        return $output\_img$
    else
        return img
    end if
end function