SOFT RANKING THRESHOLD LOSSES FOR IMAGE RETRIEVAL

Chiao-An Yang[1] *Zhixiang Wang*[1] *Yen-Yu Lin*[2] *Yung-Yu Chuang*[1]

[1]National Taiwan University [2] National Chiao Tung University

ABSTRACT

This paper proposes a novel loss, soft ranking threshold loss, for driving deep networks to learn better representations for image retrieval. Instead of working in the metric space, our loss works in the rank space which has a more uniform distribution and explicit scale and bounds. Our loss reduces the ranks of the distances between anchor-positive pairs below the threshold while increasing the ones between anchor-negative pairs above the threshold. In addition to the basic form, two extensions are proposed for improving the effectiveness: hard thresholds and ranking margin. Experiments show that the proposed loss outperforms the state-of-the-art losses on image retrieval applications.

***Index Terms*—** Image and video retrieval, deep learning

1. INTRODUCTION

Content-based image retrieval (CBIR) has been a popular topic in the computer vision and multimedia community. Given a gallery image set and a query image, the goal is to find images in the gallery that have the same class label as the query image. A common practice for CBIR is to first project all images from high-dimensional space to a low-dimensional feature space (*i.e.*, feature extraction). Then, use a distance metric to measure the distance between query and gallery features. Finally, output a sorted/ranked list of the gallery.

The recent booming deep learning technology offers a powerful tool for feature extraction. But its performance relies heavily on loss functions. Quite a few losses have been proposed for using deep learning in image retrieval tasks, such as triplet loss [1], center loss [2], additive angular margin loss [3] for face recognition, and batch-hard triplet loss [4], quadruplet loss [5] for person re-identification (reID). Notably, most of these losses operate in the distance space by maintaining certain properties, *e.g.*, keep distances to be less or greater than a pre-selected threshold. It can be problematic to use a pre-selected threshold since the distribution of distances can vary from batches to batches (Fig. 1).

Solutions to the problem can be categorized into two folds. One fold is to introduce *adaptive* hyper-parameters [6], while the other fold [7, 8, 9, 10, 11, 12, 13, 14], is to explore the information of *ranking*. The *ranking* is the value that shows how an element is related to others in a list or set. Compared with distance, it have three advantages. **(1)** The distribution is more uniform. **(2)** The rank values have a fixed lower bound (*i.e.*, zero), and upper bound (*i.e.*, the number of the elements). It is easier to choose appropriate hyper-parameters due to their fixed range. **(3)** A single *rank* value contains more information than a single *distance* value. A distance only indicates the dissimilarity between two instances, but a ranking value puts things in context by showing how dissimilar they are compared to other pairs of instances.

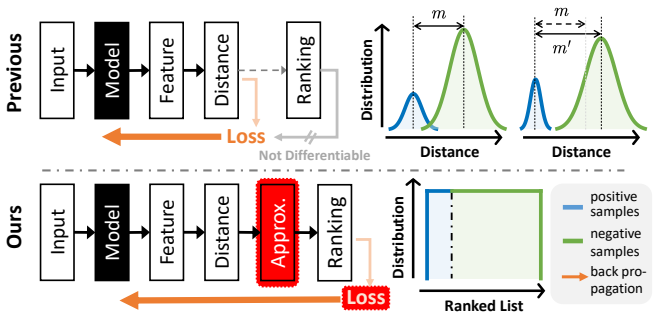


Fig. 1: Motivation. (Top) Previous deep metric learning methods, using the distance to obtain the loss, suffer from the problem that the distribution of distances varies from batches to batches. (Bottom) We propose to use the ranking representation, whose distribution is more uniform with the explicit scale and bounds, to compute loss. The differentiable soft ranking approximation makes the ranking-based loss trainable. Red boxes highlight our contributions.

Keep it in mind, we design loss functions directly using a differentiable approximation of ranking value [7], named *soft ranking threshold loss*. By introducing the mechanisms of a negative term and thresholds into, our loss can keep the rankings of positive samples below the threshold and those of negative samples above it. Two extensions are proposed to further support our ranking-based loss: hard thresholds and ranking margin. The former employs stricter thresholds for better tolerating the approximation errors of soft ranking. The latter encourages learning from more samples. Experiments show that our loss outperforms the state-of-the-art losses including the triplet loss and its variant on person reID and fashion retrieval. Chen *et al.* [13] proposed a ranking-based loss similar to ours. However, their loss only contains a positive term and the formulation is much simpler compared to ours.

The contributions of this work are highlighted as follows:

- We propose a novel loss function using the ranking as input, with both a positive and a negative term, to assert the ranking values to satisfy certain adaptive thresholds.
- We introduce the hard thresholds and ranking margin as extensions for further improving its performance.
- Experiments on CBIR applications, including person reID and fashion retrieval benchmarks, demonstrate that our loss outperforms other distance-based losses.

2. RANKING-BASED THRESHOLD LOSSES

Let's consider the scenario of image retrieval. Given an image I, assume that we have a deep model f_θ that can extract its feature vector $x = f_\theta(I)$. For a batch of B training images $\mathbf{I} = \{I_1, \cdots, I_B\}$ with corresponding labels $\mathbf{y} = \{y_1, \cdots, y_B\}$, we first extract their features $\mathbf{x} = \{x_1, \cdots, x_B\}$. For a training batch $\mathbf{x} = \{(x_i, y_i)\}^{i=1,\cdots,B}$, let $d_{ij} = \langle x_i, x_j \rangle$ be the pairwise distance between samples x_i and x_j. So, we can use d_{ij} as the element to construct the distance matrix $\mathbf{D} \in \mathbb{R}^{B\times B}$. The row vector $\mathbf{d}_i = [d_{i1}, d_{i2}, \ldots, d_{iB}]$ is the set of pairwise distances between the sample x_i and all samples in the batch. If operate the ranking function $\mathcal{R}(\bullet)$ on each element of $\mathbf{D}$, we can get the *ranking matrix* $\mathbf{R} \in \mathbb{R}^{B\times B}$, whose element $\tilde{R}_{ij} = \mathcal{R}(d_{ij}) = \#\{d_{ij} \mid d_{ij} \le d_{ik}\}$. Our goal is to derive a loss function $L(\tilde{R}_{ij})$ based on the ranking $\mathbf{R}$.

2.1. Soft ranking threshold loss

Consider the case that x_i is selected as the anchor. There exist a positive sample set $p(i) = \{j \mid y_j = y_i, i \neq j\}$ and a negative sample set $n(i) = \{j \mid y_j \neq y_i, i \neq j\}$. Let P_i and N_i be the sizes of $p(i)$ and $n(i)$ respectively. We have $P_i + N_i = B - 1$. Since positive samples should be closer to the anchor x_i and have smaller distances in $\mathbf{d}_i$, their rankings should be on the small side. Thus, we define the *hard* positive samples as those positive samples whose rankings are greater than a given threshold T_i^+. They are difficult samples because they are not ranked properly in the current feature space and more effort should be spent on them. Thus, we define the positive term of our loss function as

$$L^p = \frac{1}{P_i} \sum_{j\in p(i)} \left[R_{ij} - T_i^+\right]_+ , \tag{1}$$

where $[\bullet]_+$ is a hinge function. In this way, only hard samples (mis-ranked ones) will be penalized and the ones with larger deviation will be penalized more. Therefore, the training will focus more on correcting the difficult samples since not all data pairs are equally important to training [4, 15].

Similarly, for negative samples, we define the *hard* negative samples as those negative samples whose rankings are less than a given threshold T_i^-. For each hard negative sample, we aim to increase its ranking so that it is greater than T_i^-. Hence, the negative term of our loss is given by

$$L^n = \frac{1}{N_i} \sum_{j\in n(i)} \left[T_i^- - R_{ij}\right]_+ . \tag{2}$$

Putting the positive and negative terms together, we obtain the basic form of the proposed *soft ranking threshold loss*

$$\begin{aligned} L_{\text{SRT}} = &\alpha \overbrace{\left(\frac{1}{P_i} \sum_{j\in p(i)} \left[R_{ij} - T_i^+\right]_+\right)}^{\text{positive term}} \\ &+ (1-\alpha) \underbrace{\left(\frac{1}{N_i} \sum_{j\in n(i)} \left[T_i^- - R_{ij}\right]_+\right)}_{\text{negative term}} , \end{aligned} \tag{3}$$

where $\alpha \in [0, 1]$ is a parameter to balance the trade-off between the positive term L^p and the negative term L^n. As for the selection of the thresholds T_i^+ and T_i^-, a reasonable choice is $T_i^+ = P_i + 1$ and $T_i^- = P_i + 2$ as there are P_i positive samples in the given batch excluding the anchor x_i itself. Ideally, rankings of all positive samples should not exceed $P_i + 1$, and those of negative samples should be larger than $P_i + 1$.

However, the ranking value is not differentiate, making it impossible to directly use the output ranking of R_{ij} in our loss. In order to make L_{SRT} trainable, we approximate the hard ranking operation $\mathcal{R}(\bullet)$ using the sigmoid function

$$\tilde{R}_{ij} = \tilde{\mathcal{R}}(d_{ij}) = \sum_{k=1}^{B} \texttt{sigmoid}(d_{ij} - d_{ik}) . \tag{4}$$

Now, the loss function Eq. (3) now becomes trainable.

2.2. Hard thresholds

Using the soft ranking would introduce approximation error and the error is bounded within $(-R_{ij}/2, (B - R_{ij})/2)$. To accommodate the approximation error, we propose a stricter thresholds (*hard thresholds*) according to the bound. When considering the hard threshold for the negative term, $\hat{T}_i^-$, we hope it fulfills the requirement that for all negative samples, their soft rankings are higher than the threshold, *i.e.*, $\forall j \in n(i)$, $\tilde{R}_{ij} \ge \hat{T}_i^-$. In general, setting a higher threshold $\hat{T}_i^-$ would encourage rankings of negative samples to be higher more. However, since $\tilde{R}_{ij}$ has an upper bound, *i.e.*, $(B + R_{ij})/2$, the threshold cannot exceed it, which constraints $\hat{T}_i^-$, for all $j \in n(i)$,

$$\hat{T}_i^- \le (B + R_{ij})/2 \;\Rightarrow\; R_{ij} \ge 2\hat{T}_i^- - B . \tag{5}$$

Considering $\forall j \in n(i)$, the real ranking $R_{ij} \ge P_i + 1$, we set the hard threshold for the negative term $\hat{T}_i^-$ as $(B + P_i + 1)/2$.

The hard threshold for the positive term, $\hat{T}_i^+$, can be obtained similarly. For all positive samples x_j, $j \in p(i)$, we require them to have a ranking smaller than the threshold, *i.e.*, $\tilde{R}_{ij} \leq \hat{T}_i^+$. We would like to set the threshold as strict as possible, but it cannot be smaller than $\tilde{R}_{ij}$'s lower bound, $R_{ij}/2$. Thus, we have $\hat{T}_i^+ \geq R_{ij}/2$. Considering $\forall j \in p(i)$ the real ranking $R_{ij} \leq P_i$, we set $\hat{T}_i^+ = P_i/2$.

Theoretically, we can use the hard thresholds $\hat{T}_i^+$ and $\hat{T}_i^-$ to replace the thresholds T_i^+ and T_i^- in Eq. (3). However, empirically, we found that it is difficult for the model to get to a good starting training point and sophisticated warm-up techniques would be necessary. To prevent training from being unstable, we only apply the hard version of the loss to the batch-hardest samples after a few training iterations. We set a parameter $\beta = 0.01$ to balance the hard terms and basic terms. The full version of the proposed soft ranking threshold loss is given by $L_{\text{SRT-F}} = L_{\text{SRT}} + \beta L_{\text{SRT-H}}$, where

$$L_{\text{SRT-H}} = \overbrace{\frac{\alpha}{P_i}\left[\min_{j\in p(i)} \tilde{R}_{ij} - \hat{T}_i^+\right]_+}^{\text{positive term}} + \underbrace{\frac{(1-\alpha)}{N_i}\left[\hat{T}_i^- - \max_{j\in n(i)} \tilde{R}_{ij}\right]_+}_{\text{negative term}}. \tag{6}$$

2.3. Ranking margins

Another problem with the soft ranking threshold loss is that only a few samples contribute to the loss. Consider the example in Fig. 2(B.1), where only two "mis-ranked" samples (■ and ■) are selected and others do not contribute to the loss. Once ■ and ■ are corrected, $L_{\text{SRT}} = 0$ and the learning will stop. To have more samples contribute to the loss and learn more from samples, we introduce ranking margins.

Hard margin. By introducing a hard margin m to the thresholds, our basic loss becomes

$$L_{\text{SRT-M}} = \overbrace{\alpha\left(\frac{1}{P_i}\sum_{j\in p(i)}\left[\tilde{R}_{ij} - (T_i^+ - m)\right]_+\right)}^{\text{positive term}} + \underbrace{(1-\alpha)\left(\frac{1}{N_i}\sum_{j\in n(i)}\left[(T_i^- + m) - \tilde{R}_{ij}\right]_+\right)}_{\text{negative term}}. \tag{7}$$

Thanks to the introduced margin, our loss can force the network to keep training (Fig. 2(B.2)).

Soft margin. We use the *softplus function* $\sigma_{sp}(x)$ as an approximation of the hinge function, which is also referred as *soft margin* [4]. Unlike $\max(x, 0)$, the derivatives of $\sigma_{sp}(x)$ is not zero even when x less than zero (Fig. 2(B.3)). So, the basic loss has this form:

$$L_{\text{SRT-SM}} = \overbrace{\alpha\left(\frac{1}{P_i}\sum_{j\in p(i)}\sigma_{sp}\left(\tilde{R}_{ij} - T_i^+\right)\right)}^{\text{positive term}} + \underbrace{(1-\alpha)\left(\frac{1}{N_i}\sum_{j\in n(i)}\sigma_{sp}\left(T_i^- - \tilde{R}_{ij}\right)\right)}_{\text{negative term}}. \tag{8}$$

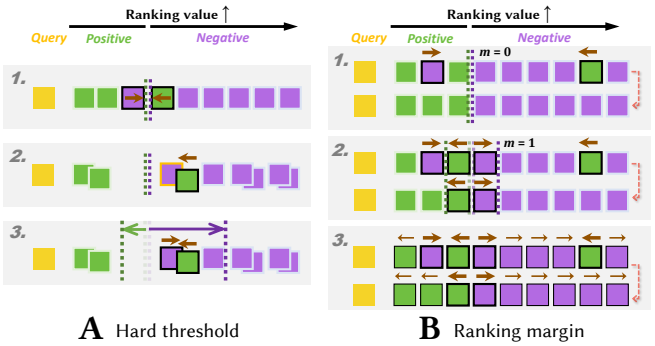


Fig. 2: (A.1) is the ideal scenario where the computed soft ranking and actual ranking are coherent, and the approximation error is small and can be ignored. □ indicate that the samples are 'chosen' by our thresholds. We can observe that all "mis-ranked" samples (■/■) are correctly chosen. (A.2) when the approximation error cannot be ignored, and the thresholds are not good enough to find all "mis-ranked" samples (neglect ■). (A.3) shows that by modifying the threshold to its hard version, this problem can be solved. (B.1) Hard margin with $m = 0$ (the basic form). Only those disarranged positive and negative samples contribute to the loss. The red arrows on the top of samples indicate gradients. Their thickness indicate the magnitudes of the gradients. After correcting the disarranged samples, the learning stops since the loss is zero. (B.2) Hard margin with $m = 1$. In addition to the disarranged samples, samples near the boundary also contribute to the loss. (B.3) Soft margin. All samples contribute to the loss depending how badly they are disarranged.

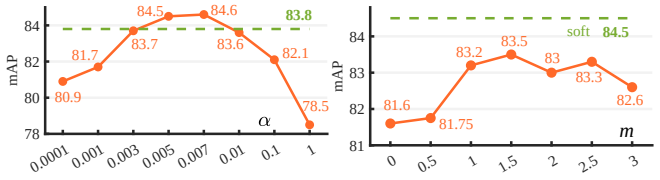
3. EXPERIMENTS

3.1. Datasets and evaluation metrics

We conduct experiments on four popular image retrieval datasets: three for person reID, including Market1501 [16], CUHK03 [17], and DukeMTMC [18] and one for fashion retrieval, *i.e.*, DeepFashion [19]. We use the mean average precision score (mAP) and the cumulative matching curve (CMC) as the metrics to evaluate performance.

3.2. Compared losses

The goal of the work is to propose a new ranking-based loss for improving retrieval performance, rather than reaching the



(a) Study of α (b) Study of m

Basic form	+Hard threshold	+Ranking margin	Full
81.6	81.8	84.5	**84.7**

(c) Ablation study

Fig. 3: Performance study.

state-of-the-art performance on individual applications. Thus, we only compare to other designs of losses for improving retrieval performance, including *triplet loss* [1], *batch hard triplet loss* [4], *hard negative triplet loss* [20] and *adaptive weighted triplet loss* [15].

3.3. Implementation details

Following [4, 15], we choose ResNet-50-v1[21] as the backbone. We discard the last softmax layer, then add one max pooling layer, a 1024-dim fully connected layer followed by a 128-dim one at the end. The batch sampling is performed in the same way as the previous work [4], where each batch has the size of 72, containing 18 persons (for person ReID) or clothes items (for fashion retrieval) with 4 images each. We also implement random erase augmentation [22] and hard identity mining (HIM) [15] with the pooling size set to 50.

3.4. Parameter and ablation study

To study the performance of our loss under different settings, we split the original training set of Market1501 into training and validation set, with the split ratio of 7:3. All results in this section are reported based on Market1501-validation.

Fig. 3(a) shows the impact of the parameter α on the performance. The result demonstrates that both the positive and negative terms in our loss are important. The model does not learn well when one of the two terms becomes too dominant. Fig. 3(b) shows the impact of the rank margin m on the performance. With the introduction of non-zero rank margin to L_{SRT}, our model is generally improved by $2\% \sim 3\%$ in mAP. The green dotted line in Fig. 3(b) indicates the performance of the soft margin. It is clear that the soft margin yields the best performance. Fig. 3(c) shows that our loss performs the best when all components are included.

3.5. Comparison to state-of-the-art losses

Tab. 1 shows the comparison of our losses to the variants with the triplet loss. All losses are trained with the same backbone Resnet-v1-50 under the same training settings. Our soft ranking threshold losses outperform all members in the triplet loss family by 3% to 5% in mAP on the person reID dataset.

For fashion retrieval, Tab. 2 shows that our soft ranking threshold loss is better than or comparable to most other losses except for the hard negative triplet loss. However, the full version of our loss (SRT-F) outperforms all losses. In addition, note that batch hard triplet loss only performs well on cross-domain fashion retrieval, but fails on in-domain tasks, while our loss performs well on both tasks.

Table 1: Comparisons with other losses on person ReID datasets, Market1501, CUHK03-NP, and DukeMTMC. The asterisk (*) indicates that the hard mining technique is used during training. SRT-F denotes our full loss by including all extensions.

Loss Type	Market1501		CUHK03-NP detected		CUHK03-NP labeled		DukeMTMC	
	mAP	CMC@1	mAP	CMC@1	mAP	CMC@1	mAP	CMC@1
Softmax	53.8	79.2	23.6	28.9	27.9	31.8	47.0	68.3
Tri	68.7	85.0	52.4	58.9	56.4	62.9	59.1	76.2
Tri-HN	73.0	87.9	55.3	61.4	58.4	65.1	62.9	79.7
Tri-BH	74.0	88.5	56.0	59.4	58.9	64.4	63.6	78.9
Tri-AW	75.3	89.4	58.2	64.0	60.7	66.9	64.1	80.2
Tri-AW*	76.5	89.7	58.9	64.5	61.1	67.1	65.5	81.4
SRT	77.3	90.1	59.4	65.3	62.9	68.6	65.9	81.1
SRT-F	78.6	90.3	62.1	67.5	65.1	70.5	**68.3**	82.3
SRT-F*	**79.2**	**90.8**	**63.0**	**68.1**	**65.8**	**71.3**	67.9	**83.0**

Table 2: Comparisons with other losses on DeepFashion. $\times$ indicates that the task is too difficult for the loss to perform well.

Loss Type	Consumer-to-shop w/o bbox		Consumer-to-shop w/ bbox		In-shop w/o bbox		In-shop w/ bbox	
	mAP	CMC@20	mAP	CMC@20	mAP	CMC@1	mAP	CMC@1
Softmax	$\times$	$\times$	$\times$	$\times$	53.0	73.2	51.3	71.2
Tri.	13.8	45.0	20.2	57.0	65.4	81.6	65.5	81.8
Tri-HN	20.6	56.5	28.8	68.5	69.2	85.4	68.1	84.4
Tri-BH	$\times$	$\times$	26.3	64.1	69.3	85.2	68.1	83.5
Tri-AW	19.7	55.8	27.1	66.9	70.4	85.9	70.3	85.8
SRT	19.4	55.1	27.3	67.0	71.4	**87.2**	**71.4**	**86.9**
SRT-F	**21.2**	**58.0**	**28.9**	**68.7**	**71.6**	86.9	71.2	86.5

4. CONCLUSION

In this paper, we design a new loss functions, called soft ranking threshold loss, using the differentiable approximation of ranking directly. It minimizes the ranking values of anchor-positive pairs and maximize the ranking values of anchor-negative pairs. In addition to the basic form, we have also explored two extensions: hard thresholds and ranking margin, for further improving the performance. Experiments show that our losses outperform other losses on person ReID and fashion retrieval.

5. REFERENCES

[1] Florian Schroff, Dmitry Kalenichenko, and James Philbin, "Facenet: A unified embedding for face recognition and clustering," in *CVPR*, 2015. 1, 4

[2] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao, "A discriminative feature learning approach for deep face recognition," in *ECCV*, 2016. 1

[3] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *CVPR*, 2019. 1

[4] Alexander Hermans, Lucas Beyer, and Bastian Leibe, "In defense of the triplet loss for person re-identification," *arXiv preprint arXiv:1703.07737*, 2017. 1, 2, 3, 4

[5] Weihua Chen, Xiaotang Chen, Jianguo Zhang, and Kaiqi Huang, "Beyond triplet loss: a deep quadruplet network for person re-identification," in *CVPR*, 2017. 1

[6] Qiqi Xiao, Hao Luo, and Chi Zhang, "Margin sample mining loss: A deep learning based method for person re-identification," *arXiv preprint arXiv:1710.00478*, 2017. 1

[7] Lian Yan, Robert H Dodier, Michael Mozer, and Richard H Wolniewicz, "Optimizing classifier performance via an approximation to the wilcoxon-mann-whitney statistic," in *ICML*, 2003. 1

[8] Qifan Wang, Zhiwei Zhang, and Luo Si, "Ranking preserving hashing for fast similarity search," in *IJCAI*, 2015. 1

[9] Jerome Revaud, Jon Almazán, Rafael S Rezende, and Cesar Roberto de Souza, "Learning with average precision: Training image retrieval with a listwise loss," in *ICCV*, 2019. 1

[10] Jun Wang, Wei Liu, Andy X Sun, and Yu-Gang Jiang, "Learning hash codes with listwise supervision," in *ICCV*, 2013. 1

[11] Svebor Karaman, Xudong Lin, Xuefeng Hu, and Shih-Fu Chang, "Unsupervised rank-preserving hashing for large-scale image retrieval," in *Proceedings of the 2019 on International Conference on Multimedia Retrieval*, 2019. 1

[12] Jin Wang, Zheng Wang, Changxin Gao, Nong Sang, and Rui Huang, "Deeplist: Learning deep features with adaptive listwise constraint for person reidentification," *IEEE Transactions on Circuits and Systems for Video Technology*, 2016. 1

[13] Shi-Zhe Chen, Chun-Chao Guo, and Jian-Huang Lai, "Deep ranking for person re-identification via joint representation learning," *TIP*, vol. 25, no. 5, 2016. 1

[14] Xinshao Wang, Yang Hua, Elyor Kodirov, Guosheng Hu, Romain Garnier, and Neil M Robertson, "Ranked list loss for deep metric learning," *arXiv preprint arXiv:1903.03238*, 2019. 1

[15] Ergys Ristani and Carlo Tomasi, "Features for multi-target multi-camera tracking and re-identification," in *CVPR*, 2018. 2, 4

[16] Liang Zheng, Liyue Shen, Lu Tian, Shengjin Wang, Jingdong Wang, and Qi Tian, "Scalable person re-identification: A benchmark," in *ICCV*, 2015. 3

[17] Wei Li, Rui Zhao, Tong Xiao, and Xiaogang Wang, "Deepreid: Deep filter pairing neural network for person re-identification," in *CVPR*, 2014. 3

[18] Ergys Ristani, Francesco Solera, Roger Zou, Rita Cucchiara, and Carlo Tomasi, "Performance measures and a data set for multi-target, multi-camera tracking," in *ECCV Workshop on Benchmarking Multi-Target Tracking*, 2016. 3

[19] Ziwei Liu, Ping Luo, Shi Qiu, Xiaogang Wang, and Xiaoou Tang, "Deepfashion: Powering robust clothes recognition and retrieval with rich annotations," in *CVPR*, June 2016. 3

[20] Eric Dodds, Huy Nguyen, Simao Herdade, Jack Culpepper, Andrew Kae, and Pierre Garrigues, "Learning embeddings for product visual search with triplet loss and online sampling," *arXiv preprint arXiv:1810.04652*, 2018. 4

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *CVPR*, 2016. 4

[22] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang, "Random erasing data augmentation," *arXiv preprint arXiv:1708.04896*, 2017. 4