Negative Samples are at Large: Leveraging Hard-distance Elastic Loss for Re-identification

Hyungtae Lee[1] Sungmin Eum[1,2] Heesung Kwon[1]

[1]DEVCOM Army Research Laboratory [2]Booz Allen Hamilton

Abstract. We present a Momentum Re-identification (MoReID) framework that can leverage a very large number of negative samples in training for general re-identification task. The design of this framework is inspired by Momentum Contrast (MoCo), which uses a dictionary to store current and past batches to build a large set of encoded samples. As we find it less effective to use past positive samples which may be highly inconsistent to the encoded feature property formed with the current positive samples, MoReID is designed to use only a large number of negative samples stored in the dictionary. However, if we train the model using the widely used Triplet loss that uses only one sample to represent a set of positive/negative samples, it is hard to effectively leverage the enlarged set of negative samples acquired by the MoReID framework. To maximize the advantage of using the scaled-up negative sample set, we newly introduce Hard-distance Elastic loss (HE loss), which is capable of using more than one hard sample to represent a large number of samples. Our experiments demonstrate that a large number of negative samples provided by MoReID framework can be utilized at full capacity only with the HE loss, achieving the state-of-the-art accuracy on three re-ID benchmarks, VeRi-776, Market-1501, and VeRi-Wild.

Keywords: Re-identification, large-scale negative samples, momentum encoder, MoReID, HE loss

1 Introduction

Re-identification (re-ID) is the task of finding instances (e.g., person, vehicle) with the same identity in images taken from different viewpoints in different locations. In general, a re-ID model needs the ability that, given a query, well separates instances with the same ID (positive samples) from instances with different IDs (negative samples). This ability can be acquired by training the model so that the positive samples are placed closer to the query while the negative samples are placed in a distant location in the learned feature space.

When separating the positives from the negatives with respect to a given query, such as in the re-ID, having a sufficient number of samples in each set often helps in generating a more precise boundary between the two sets. However, the mini-batch optimization which is widely used in training a network cannot afford to use more than a certain number of samples in each training

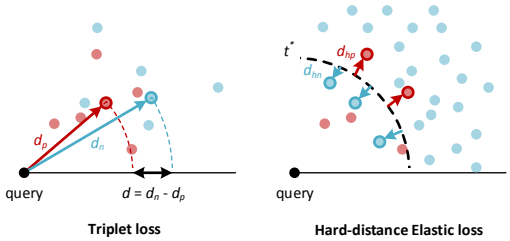


Fig. 1: **Hard-distance Elastic Loss vs. Triplet loss.** In Triplet loss, the positive and negative samples are each represented by a single point. HE loss, on the other hand, can use more than one hard samples to represent positive and negative samples, respectively. Knowing that the optimally determined boundary t^* divides the feature space into positive and negative sides, a positive sample is a called a hard sample when it lies on the negative side, and vice versa. Representative samples are plotted in bold. Multiple points are better than a single point when representing a large number of negative samples, as shown in the figure on the right. More analysis of HE loss versus Triplet loss is in Section 4.4.

iteration. Recently, Momentum Contrast (MoCo) [12] provided a structural way to effectively provide a large number of samples for training. It uses a dictionary that collects several preceding batches in the form of encoded features. To fully take advantage of using a large number of samples when generating the precise boundary, features of these samples are desired to be encoded in a consistent feature space. To maintain this consistency as training progresses, MoCo uses an additional slow-progressing encoder that only processes samples to be stored in the dictionary (these samples are called "keys" in [12]). The slow-progressing property of the additional encoder is acquired by updating via a moving average from the main encoder with a large momentum.

To enable the usage of a large number of negative samples for training a re-ID model, we devise Momentum ReID (MoReID) architecture that is constructed with a Siamese network and a dictionary from MoCo. During training, a batch consisting of multiple groups, where the samples in the same group share the same ID, is fed to both networks. Only the outputs of the secondary *forward-only* encoder with their corresponding IDs are stored in the dictionary. We treat each encoded sample from the main network as the query, which is compared with each of the keys in the dictionary to validate whether they match or not. All the keys with IDs different from the query are considered as negative samples. Among the keys that share the same ID with the query, only the samples that just came in from the current batch are used as positive samples (i.e., present positives) for training. Past positives are not used. While processing a batch of 256 samples, MoReID can leverage up to 8192 negative samples accumulated within the dictionary when using 4 GPUs with 48G GPU memory.

To take full advantage of a very large number of negative samples accumulated within the MoReID architecture, we use a novel loss called *Hard-distance Elastic Loss* (HE Loss). Previous re-ID methods have mainly used the Triplet loss [16, 29, 15] which computes the difference between the distance to a single

positive representation and the distance to a single negative representation for a given query (left illustration of Figure 1). A single representation is an average feature over all samples or hard samples in each group. As the number of samples increases, the representativeness of a single representation gets less accurate. Because of this, it does not scale well with the sample size.

Instead of treating a group of samples with a single representation, the HE loss has the capability of leveraging multiple hard samples that are on the other side of the boundary optimally drawn between the positive and negative samples (i.e., positives on the negative side and negatives on the positive side). In detail, the HE loss is calculated as the sum of the penalties, which is the extent to which the boundary is crossed for all hard samples (right illustration of Figure 1). For each query, the boundary is optimally determined to minimize the loss. We have theoretically demonstrated that it is optimal to place the boundary in the region where the number of hard positive samples and the number of hard negative samples are equal.

To demonstrate the effectiveness of the MoReID architecture and the HE loss, we carried out experiments on person re-ID and vehicle re-ID tasks. Experiments show that deploying only one of the two modules does not bring any advantage in improving re-ID accuracy. However, using the two modules together provides drastic improvement in accuracy when compared to the baseline without the modules in all re-ID tasks. The two novel modules work mutually and in a complementary way, meaning that the large negative training samples acquired using the MoReID architecture cannot be used effectively without the HE loss, whereas HE loss can only demonstrate its full strength when the number of 'negative samples are at large'.

2 Related Works

ReID loss. Existing re-ID methods typically use two types of losses with different label forms: pairwise (i.e., class-level label and model estimate) loss and tripletwise (i.e., query, and positive and negative samples for the query) loss. A pairwise loss defined to minimize the difference between the label and estimate is mainly used for ID classification in re-ID methods. Tripletwise loss is used in re-ID methods to bring samples with the same ID closer together and push the samples with different IDs further away from each other.

For the pairwise loss, there are several commonly used losses, e.g., softmax [33, 35], cross-entropy loss [3, 11, 22, 27], cosface [35], arcface loss [7], and circle loss [31, 43]. On the other hand, Triplet loss [15] is the most representative of the tripletwise loss in the re-ID methods. Triplet loss measures the relationship between a query sample and two single representative samples representing the positives and the negatives, respectively. Each representation can be calculated using either a weighted sum of all samples in each group [11, 17] or hard sample mining [41, 31, 1, 22, 27]. However, as the number of samples increases significantly, it becomes more difficult to adequately represent them with a single representation, which we call the sample scalability issue. Other metric

learning losses that can be used as a tripletwise loss also face similar limitations, e.g., c-triplet loss [34], circle loss [31, 43], CDF-based weighted triplet loss [41], etc. To address sample scalability, N-pair loss [30] and Ranked List loss [37] are defined to use multiple hard samples as representative samples, but show lower re-ID performance than Triplet loss because the number of hard samples is not optimally determined. In this paper, we introduce a novel loss, referred to as *Hard-distance Elastic Loss* (HE Loss), that can better cope with the sample scalability issue to exhibit significantly higher accuracy than the Triplet loss.

Exploring a large number of negative samples. Recently, exploration of a very large number of negative samples in training has shown promising accuracy in various unsupervised representation learning tasks. Most of the methods presenting such promising accuracy employ the MoCo framework [12], which provides a structural way to generate a large number of negative samples. This MoCo framework has also been used for several Re-ID methods, but is only limited to unsupervised learning settings [2, 8, 44, 46]. For the task of supervised re-ID, our MoReID is the first architecture to adopt MoCo to leverage a large number of negative samples with the help of the proposed HE loss.

3 Method

3.1 Hard-distance Elastic Loss

Definition of HE loss. Hard-distance Elastic (HE) loss measures the degree to which samples cross the boundary (i.e., positive samples located on the negative side, and vice versa) that is determined to divide the feature space into positive and negative regions. Here, samples in the opposite side of the boundary are considered hard samples.

For a query q, key samples are split into its positives $p \in P$ and its negatives $n \in N$ according to their IDs, where P and N are the positive and negative sets, respectively. HE loss, $\mathcal{L}_q$, is defined with the boundary t as follows:

$$\mathcal{L}_q(t) = \sum_{p \in P} \max(d_{pq} - t, 0) + \sum_{n \in N} \max(t - d_{nq}, 0), \tag{1}$$

where d_{pq} and d_{nq} are the distances (e.g., Euclidean distance in our experiments) from q to p and from q to n, respectively. The optimal boundary t^* can be acquired by minimizing the HE loss as follows:

$$t^* = \arg\min_t \mathcal{L}_q(t). \tag{2}$$

Derivation of the optimal boundary. The optimal point of the boundary to minimize the HE loss must satisfy two conditions: i) the derivative of the HE loss with respect to the boundary t is zero at this optimal point, and ii) the HE loss is convex with respect to the boundary t. The derivative of $\mathcal{L}_q(t)$ with respect to the boundary t can be derived from eq. 1 as follows:

$$\frac{d\mathcal{L}_q(t)}{dt} = \sum_{p \in P} -\mathbb{1}(d_{pq} > t) + \sum_{n \in N} \mathbb{1}(d_{nq} < t) = -N_{hp}(t) + N_{hn}(t) \tag{3}$$

where $\mathbb{1}(\cdot)$ is a unit function. $N_{hp}(t)$ and $N_{hn}(t)$ are the number of hard positive samples and hard negative samples, respectively, when the boundary is t. Note that $\mathbb{1}(d_{pq} > t)$ or $\mathbb{1}(d_{nq} < t)$ represent samples that are located on the other side of the boundary with respect to its identity. In other words, the samples satisfying the constraints within the unit functions in eq. 3 are the hard samples. Accordingly, the first condition to define an optimal boundary is satisfied when the number of hard positive samples and the number of hard negative samples are equal (i.e., $N_{hp} = N_{hn}$).

In addition, $N_{hp}(t)$ is a monotonically decreasing function for t as the number of hard positive samples decreases as the boundary moves away from the query. Similarly, $N_{hn}(t)$ is a monotonically increasing function. Accordingly, $d\mathcal{L}_q(t)/dt$ in eq. 3 is a monotonically increasing function and the HE loss is convex with respect to t, satisfying the second condition for an optimal boundary.

Knowing that the second condition is always met, the task of finding the optimal boundary is narrowed down to localizing the point where $N_{hp} = N_{hn}$. We first compute the distances from all the samples to the query and sort them in ascending order. Then the shortest distance is set to be the first boundary candidate to evaluate whether $N_{hp} = N_{hn}$ is satisfied. If satisfied, the candidate is set as t^*. If not, we iterate through the sorted distances until the condition is met. HE loss for a given batch is acquired by processing all the queries and their optimal boundaries independently and averaging the results. Pseudo-code for computing the optimal boundary and the HE loss is included in Algorithm 1.

3.2 MoReID

Architecture. A core component of MoReID is being able to use a very large number of negative samples in training a re-ID model. To enable this capability, MoReID leverages the MoCo architecture [12], which is equipped with a dictionary that continuously collects a large number of encoded features for incoming input images as the training evolves. The dictionary is a queue where the oldest batch leave as a new batch comes in.

To take full advantage of the benefits of using a large number of samples (especially, when generating the precise boundary between positive and negative samples in the feature space), it is desirable that these samples be encoded into a consistent feature space. To this end, MoCo uses a separate, slow-progressing encoder in addition to the main encoder to encode the samples to lie in a nearly consistent feature space. Note that this slow-progressing encoder only encodes the key samples that are fed into the dictionary.

While the main encoder is updated via back-propagation, this slow-progressing encoder ("forward-only encoder" in figure 2) is updated via a moving average with momentum m from the main encoder (i.e., $f_k = m \cdot f_k + (1 - m) \cdot f_q$, where f_k and f_q are the slow-progressing encoder and the main encoder, respectively). The slow progress property in the forward-only encoder is acquired by reducing the extent of update with a large momentum m. Note that the forward-only encoder is only used for training, as its purpose is to provide a large number

of samples (negative samples for our method) to be compared against a given query.

As a result, our MoReID architecture is also designed as a Siamese network that consists of the main encoder and the additional forward-only encoder. Both encoders take the same image batch as input where each image batch consists of C groups, and each group contains N images with the same ID. The main encoder is trained with two losses: pairwise ID loss and the proposed HE loss. For ID classification, the output of the main encoder is fed through an ID prediction layer which generates the input to the computation of the ID loss. We use circle loss [31] to serve as the ID loss[1]. The proposed MoReID structure is shown in Figure 2.

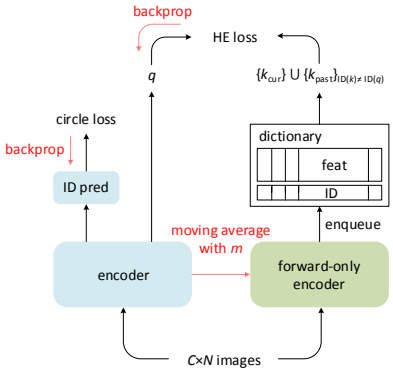


Fig. 2: **MoReID.** Black arrows and red arrows represent forward computations and model updates, respectively.

Labeling key samples in the dictionary. Within the dictionary, only the samples that do not share the same ID with a given query are treated as negative samples. As for the positive samples (samples with same ID w.r.t the query), only the samples from the current batch is considered, while disregarding the ones that had been previously stored in the dictionary. This was an empirical decision. The experiments demonstrated that leaving out the past positive samples led to higher accuracy, and details on this are found in the supplementary material. If a model is trained with past positive samples, the model is less likely to carry the capability to properly separate the positives from the negatives in the current feature space. This may be because the past positive samples do not share the consistent feature space with the query. As no past positive samples are used for training, only *negative samples are at large*.

As the samples within the dictionary are constantly picked out based on their IDs, the dictionary is constructed so as to contain both the features and their IDs. The `LabelDictionary` function in Algorithm 1 deals with the labeling process (positive or negative w.r.t. the query) described in this section.

3.3 Implementation Details

Encoder backbone. For the encoder backbone design, we follow the optimal configuration of a Re-ID network in [14]. ResNet-50 [13] is used which is followed by two non-local modules [36]. The output from the non-local modules are pooled by the generalized mean pooling (i.e., $\left(1/|\mathbf{X}|\sum_{x\in\mathbf{X}} x^{\alpha}\right)^{1/\alpha}$, where $\alpha = 3$ in our experiments). All first batch normalizations in each residual module

[1] Circle loss [31] is designed in two types, pairwise and tripletwise, and we adopt the pairwise type as the ID loss in MoReID architecture.

Algorithm 1: MoReID: PyTorch-like Pseudo-code

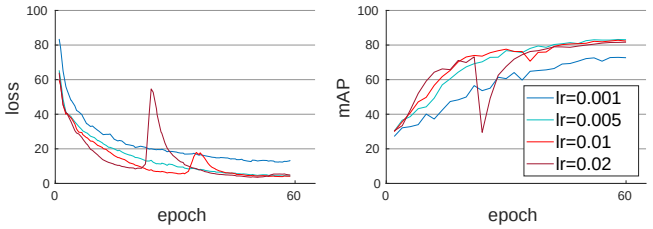
```
# f_q, f_k: main and secondary encoders
# h: ID pred. layer
# feat, id: dictionary for feat and ID
# m: momentum

f_k.params = f_q.params # initialize
# load a minibatch (image and label)
for x, l in loader:
    Q, K = f_q(x), f_k(x) # encode
    P = h(Q) # predict ID
    K = K.detach() # no gradient to f_k

    dequeue(feat) # dequeue the earliest feat
    enqueue(feat, K) # enqueue the cur feat
    dequeue(id) # dequeue the earliest id
    enqueue(id, l) # enqueue the cur id

    # loss (closs: circle loss)
    loss = heloss(Q, feat, l, id) + closs(P, l)

    # update MoReID
    loss.backward()
    f_k.params = m*f_k.params
                 +(1-m)*f_q.params
```

```
# HE loss
def heloss(Q, K, id_Q, id_K):
    loss = 0
    for q, id_q in zip(Q, id_Q):
        # label key samples in the dictionary
        # P, N: positive and negative keys
        P, N = LabelDictionary(K, id_K, id_q)
        n_p = P.size(0) # num of positives
        # calculate euclidean distance
        d_pq, d_nq = eucl(q, P), eucl(q, N)
        # sort in ascending order
        d_pq, d_nq = d_pq.sort(), d_nq.sort()

        # search the optimal boundary
        # n_hp, n_hn: num of hard samples
        n_hp, n_hn = n_p, 0
        d_pq_cur, d_nq_cur = d_pq[0], d_nq[0]
        while 1:
            if d_pq_cur <= d_nq_cur:
                t = d_pq_cur
                n_hp -= 1
                d_pq_cur = d_pq[n_p-n_hp]
            else:
                t = d_nq_cur
                n_hn += 1
                d_nq_cur = d_pq[n_hn]

            if n_hp == n_hn: break # terminate

        # calculate loss
        loss += relu(d_pq - t).sum()
                    + relu(t - d_nq).sum()
    return loss/Q.size(0)
```

have been replaced by instance batch normalizations [32]. The backbone network was initialized on ImageNet-pretrained ResNet-50. For ID prediction, one fully connected layer is used.

Inference. The re-ID task is to retrieve matching samples from the gallery for a given query. For the inference, the only main encoder is used to encode both query and gallery samples in the learned feature space. The similarity between a pair of encoded samples (query and gallery) is measured with the Euclidean distance.

Optimization. We use SGD as our optimizer. Weight decay and momentum are 0.0005 and 0.9, respectively. Base learning rate was set based on the dataset. We have conducted a study in the experiment section where we investigate the relationship between the dataset and the base learning rates. (see Sec. 4.1). The training schedule is commonly designed independent of the dataset so that the learning rate gradually decays to zero at the cosine rule beginning at the middle of the total training epoch. This is to gradually reduce the difference between the parameters of the main encoder and the forward-only encoder while training. Once the training is done, the key samples in the dictionary can be compared with their query sample in the 'close-to-equivalent' feature space.



lr	mAP
0.001	72.7
0.005	83.1
0.01	**83.3**
0.02	81.7

Table 1: **Comparisons of different learning rates.** The figures on the left and middle show the loss evolution and accuracy trend as training progresses, respectively. The table on the right shows mAPs of the models trained with different learning rates.

4 Experiments

To confirm the effectiveness of the proposed re-ID method, we use three re-ID datasets: VeRi-776, Market-1501, and VeRi-Wild. Unless specified, all experiments (e.g., ablation studies, comparison of HE loss to other losses, etc.) were carried out on VeRi-776 dataset. Comparing our method with the state-of-the-art methods is performed on all three re-ID datasets. mAP and R-1 are commonly used as evaluation metrics for all the datasets.

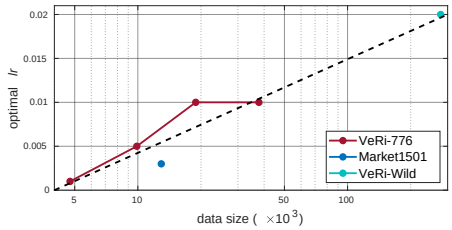
Training schedule. For VeRi-776 and Market-1501, we trained for 60 epochs. For the large-scale VeRi-Wild dataset, we trained for 120 epochs.

Pre-processing. For training, three pre-processing approaches were found to be optimal after evaluating all possible combinations of multiple approaches in [14]: horizontal flipping, random erasing [47], and auto-augment [6]. For the vehicle re-ID datasets (i.e., VeRi-776 and VeRi-Wild), the input images are resized to 256×256. For the person re-ID dataset (i.e., Market-1501), the input image is resized to 384 ×128.

4.1 Study for Optimal Learning Rate

Searching the optimal learning rate. To search for the optimal learning rate for MoReID, we compare models trained with different learning rates as shown in Table 1. The optimal learning rate with respect to accuracy was 0.01. When the learning rate is lower (i.e., 0.001) than this optimal rate, the model was under-fitted, converging at a relatively higher loss with a degraded accuracy.

On the other hand, when the learning rate was higher than or equal to the optimal rate, the training seemed unstable as a spike occurs in the middle of the training (Left figure in Table 1). The spike triggered a sudden drop in accuracy (Middle figure in Table 1). This spike in loss was also observed in [5] when there was a sudden change in gradient at the initial layer of the model. Based on this observation, [5] provided a temporary solution for this instability by freezing the initial layer, which led to an increased accuracy. However, suppressing the gradient of the initial layer, including this layer freezing, gradient clipping, etc.,



	data size	lr_s	mAP
subsets	4,751	0.001	40.9
	9,889	0.005	56.7
	18,882	0.01	70.3
entire	37,775	0.01	83.3

Table 2: **The optimal learning rate with respect to the size of the dataset.** The data size is expressed on a logarithmic scale along the x coordinate in the figure on the left. The dashed line in the figure is drawn to represent the approximate relationship between the optimal learning rate and the data size, expressed in eq. 4. The data size, the optimal learning rate and accuracy for VeRi-776 and its three subsets are shown in the table on the right.

was found to be ineffective in dealing with the instability within our scenario. More studies are needed to address this unique training instability in future works.

The optimal learning rate with respect to the dataset. As the accuracy of our method was somewhat sensitive to the learning rate, we saw the need to provide a study that shows the relationship between the optimal learning rate and a given dataset (especially the size of the dataset). This would prevent having to go through an exhaustive search just to find out the optimal learning rate when applying our method to a different dataset.

To figure out this relationship within the same dataset, we generate three subsets from VeRi-776 dataset by collecting samples each associated with 1/2, 1/4, and 1/8 of the entire IDs. Then, we check whether the relationship between the optimal learning rates of the entire VeRi-776 dataset and the three subsets were consistently applicable when we trained on other datasets (i.e., Market-1501 and VeRi-Wild). As shown in the figure of Table 2, the optimal learning rate should increase logarithmically as the size of the dataset size gets larger and the relationship can simply be expressed as follows:

$$\begin{aligned} lr_s &= 0.02\frac{\log(s) - \log(4\times10^3)}{\log(3\times10^5) - \log(4\times10^3)} \\ &\propto \log(s) \quad \text{if} \quad s \gg 4\times10^3, \end{aligned} \tag{4}$$

where s is the size of the dataset. Note that $\log(3\times10^5)$ and 0.02 in eq. 4 are the size of the largest dataset (i.e., VeRi-Wild) used to derive this equation and the optimal learning rate for this dataset, respectively. lr_s is a reference value that can be used as the optimal learning rate when applying our method to a different dataset. From these observations, we can set a rule for searching the optimal lr with respect to the dataset size as:

"When the dataset size is increased by power of k, multiply the learning rate by k".

As this rule does not conflict with the linear scale rule [9] that defines the optimal lr with respect to the batch size, these two rules can be applied concurrently.

4.2 Ablation Studies

Momentum m. We compare the accuracy of models trained with different momentum m as shown in the table below:

	no dict.	m 0.9	0.99	0.995	0.997	0.999
mAP	80.6	79.5	80.7	82.4	**83.3**	83.0
gain	·	(-1.1)	(+0.1)	(+1.8)	(+2.7)	(+2.4)

When m is ≥ 0.995, the accuracy was reasonably high. This indicates that maintaining the key samples consistent by slowly updating the forward-only encoder was critical to achieving high accuracy. It is noteworthy that when consistency of samples was less maintained (i.e., $m \leq 0.9$), there was even a loss in accuracy when compared to "no dict.".

Dictionary size. We experiment how different dictionary sizes affect the mAPs. This comparison is shown in the table below:

	no dict.	dict. size 1024	2048	4096	8192
# batch	·	4	8	16	32
optimal m	·	0.99	0.995	0.997	0.997
mAP	80.6	81.9	82.5	83.0	**83.3**
gain	·	(+1.3)	(+1.9)	(+2.4)	(+2.7)

We have used a minibatch consisting of 256 images (C: 16, N: 16). "# batch" is the number of batches constantly stacked in a dictionary. Within our computing environment (4×12GB GPUs), the maximum dictionary size is 8192. "gain" shows how much accuracy is gained by using a dictionary when compared with a model without a dictionary ("no dict."). Regardless of the dictionary size, using a dictionary itself always provided better accuracy. In addition, increasing the dictionary size consistently bumped up the accuracy.

How long a batch is kept in the dictionary affects the optimal m that controls the speed of update for the forward-only encoder. More specifically, a slower update for the forward-only encoder (i.e., bigger m) may be needed to increase how long the past batches reside in the dictionary. We found the optimal m for each dictionary size as shown in the table above. We have experimentally validated that as the dictionary size gets larger, the optimal m also becomes larger.

Batch. HE loss is highly related to the number of hard samples. Since a large number of negative samples are used, the maximum number of hard samples actually becomes the twice the number of positive samples at the optimal boundary (i,e, $N_{hp} = N_{hn}$ in eq. 3). As each batch is fixed with the configuration of $C \times N$ (C: number of ID types, N: number of instances per ID type), the number of positive samples is also fixed as N-1. Therefore, how the batch configuration is set directly affects the HE loss with respect to the accuracy as shown below:

method	MoReID	HE loss	mAP
baseline			79.7
(a)		✓	80.6 (+0.9)
(b)	✓		79.4 (-0.3)
Ours	✓	✓	**83.3** (+3.6)

Table 3: **Accuracy with and without MoReID and HE loss, respectively.** All methods use the same training schedule. 'baseline' and 'method (a)' use a learning rate of 0.01 which has been proven to be optimal for the backbone network as in [14].

$C \times N$	128×2	64×4	32×8	16×16	8×32
mAP	77.9	79.7	83.0	**83.3**	83.2

As shown in the table, the accuracy improves when N is larger than or equal to 8. We use the optimal configuration (16×16) throughout all experiments.

4.3 Main Results

Synergy of HE loss and MoReID architecture. Here, we demonstrate how each of the two new components impact accuracy and whether they are effectively complementary to each other. In Table 3, we compare the accuracy of four different combinations with and without each component on top of the baseline. The baseline uses the same backbone network as ours, while Triplet loss are used instead of the HE loss. While using HE loss (method (a)) slightly improves the baseline accuracy by 0.9, introducing a large number of negative samples using MoReID architecture (method (b)) rather brings a degraded accuracy. When the model is equipped with both of the components, a synergistic effect can be observed where the accuracy is significantly increased over the baseline.

The result can be interpreted in three aspects: i) Being able to represent positive or negative groups with multiple hard samples (i.e., using the HE loss) has a benefit over using a single representation for each group, ii) Without a proper loss design, using a large number of samples has very little impact on accuracy improvement, and that iii) the two components effectively complement each other.

Comparison to the state-of-the-arts. Table 4 shows the results on three re-ID datasets. Our method consistently outperforms the baseline and all previous methods on all three datasets in terms of both mAP and R-1.

On **VeRi-776**, our method increases baseline accuracy by 3.3 in mAP and 0.5 in R-1. Our method also presents a better mAP by 0.2 while yielding a comparable R-1 compared to the SOTA (i.e., HRC [43]).

On **Market-1501**, we increases baseline accuracy by 2.6 in mAP and 0.8 in R-1. Our method outperforms the SOTA in mAP by 0.5 and presents a comparable R-1 performance.

On **VeRi-Wild**, our method increases baseline accuracy in all three subsets. The improvements are ∼2.6 and ∼0.6 in mAP and R-1, respectively. Our method outperforms the SOTA by ∼1.9 in mAP and by ∼3.7 in R-1, in all three subsets.

method	mAP	R-1
CAL [28]	74.3	95.4
PGAN [42]	79.3	96.5
PVEN [24]	79.5	95.6
SAVER [**?**]	79.6	96.4
HRC [43]	83.1	**97.3**
baseline	80.0	96.8
Ours	**83.3**	**97.3**
gain	+3.3	+0.5

(a) VeRi-776

method	mAP	R-1
PAT [23]	88.0	95.4
ABD [4]	88.3	95.6
BV [40]	89.1	96.0
SSGR [39]	89.3	**96.1**
CAL [28]	89.5	95.5
baseline	87.4	95.3
Ours	**90.0**	**96.1**
gain	+2.6	+0.8

(b) Market-1501

method	small		medium		large	
	mAP	R-1	mAP	R-1	mAP	R-1
BW [20]	70.5	84.2	62.8	78.2	51.6	70.0
SAVER [18]	80.9	94.5	75.3	92.7	67.7	89.5
PVEN [24]	82.5	·	77.0	·	69.7	·
PGAN [42]	83.6	95.1	78.3	92.8	70.6	89.2
HRC [43]	85.2	94.0	80.0	91.6	72.2	88.0
baseline	83.7	95.5	78.8	94.4	71.5	91.5
Ours	**86.1**	**96.1**	**81.2**	**94.5**	**74.1**	**91.7**
gain	+2.4	+0.6	+2.4	+0.1	+2.6	+0.2

(c) VeRi-Wild

Table 4: **Comparison with previous methods (top 5) on three datasets.** Numbers in the parentheses indicate the performance gains with respect to the corresponding baselines. For fair comparison, all listed methods do not use any post-processing such as re-rank nor any external datasets.

Our method consistently provides state-of-the-art accuracy for both vehicle re-ID and person re-ID tasks. This result also demonstrates that our method is scalable to a large-scale dataset (i.e., VeRi-Wild dataset). It is noteworthy to mention that the gain over the SOTA was higher on the large-scale dataset than that on other datasets.

4.4 Tripletwise ReID Loss Comparison

We compare the proposed HE loss with other tripletwise losses used to define triplet relationships beyond accuracy.

vs. Triplet loss. Figure 1 shows the conceptual difference between the HE loss and the Triplet loss. The most distinct property of HE loss compared to Triplet loss is that it can take more than one samples to represent the positive and the negative sets. As previously claimed, a single representation used by Triplet loss cannot cope well with the scalability of samples, which must be considered when using MoReID. To confirm this claim, we compare the accuracy of HE loss and Triplet loss while changing the dictionary size as shown in the Table below:

loss	dict. size				
	512	1024	2048	4096	8192
Tri-all	79.7	79.6	79.5	79.5	79.3
Tri-hard	80.2	79.9	79.7	79.5	79.7
HE loss	**81.0**	**81.9**	**82.5**	**83.0**	**83.3**

Margin of Triplet loss is set to 0.3 as in [14]. When a single representation is computed for either the positive or the negative sample set, two different Triplet losses can be deployed based on the fact whether hard sample mining precedes ("Tri-hard") or not ("Tri-all"). We can observe the benefit of using the HE loss when using a larger-scale samples, whereas using the Triplet loss does not provide such advantage. Seeing that the "Tri-hard" slightly outperforms the "Tri-all" shows that the hard sample mining, which HE loss is also inherently leveraging, is more apt in handling the scalability of samples.

There exist several Triplet loss variants which also rely on single representations for the positive and negative sets, e.g., circle triplet loss (Circle) [31],

classification version of triplet loss (C-triplet) [34], and CDF-based weighted triplet loss (W-triplet) [41]. We compare the accuracy of the proposed HE loss with these variants when used for MoReID training, as below:

Circle	C-triplet	W-triplet	**HE**
79.1	80.3	79.9	**83.3**

Dictionary size of 8192 is used consistently for all losses. HE loss presents significantly better accuracy than the other losses, demonstrating the HE loss's ability to effectively process large-scale negative samples in training.

vs. Losses that allows for multiple negatives. Some previous losses have provided a way to represent an entire set of samples with multiple samples to better cope with large-scale negative samples. For example, N-pair loss [30] can use multiple hard samples to represent all negative samples and Ranked List loss [37] can use multiple hard samples to represent both positive and negative samples. To compare HE loss with N-pair and Ranked List losses in handling the large-scale negative samples, we have set up an experiment to optimize two frameworks, the backbone network (our baseline in Table 3) and the MoReID network (dict. size is 8192). While the baseline processes all triplet combinations within the current batch only, the MoReID network also leverages past samples to be used in building a larger-scale negative sample set.

re-ID framework	Triplet	N-pair	Ranked List	**HE**
baseline	79.7	78.8	78.8	**80.6**
MoReID	79.4 (-0.3)	79.8 (+1.0)	80.5 (+1.7)	**83.3 (+2.7)**

Numbers in parentheses indicate gaps from its corresponding baseline. N-pair loss and Ranked List loss present better accuracy with MoReID than with their baselines, demonstrating their ability to properly handle large-scale negative samples. Interestingly, both losses underperformed the Triplet loss when with the baselines, but outperformed it with MoReID. Remarkably, HE loss yields the best accuracy as well as the highest margin from its baseline. It shows the superiority of HE loss with regard to re-ID performance and its ability to process large-scale negative samples.

vs. InfoNCE loss. InfoNCE [26] is a widely used loss function in self-supervised learning that has been actively studied recently. InfoNCE also involves the triplet relationships which is formulated as shown below:

$$\mathcal{L}_q^i = -\log \frac{\exp(q \cdot p/\tau)}{\sum_{k \in \mathbf{K}} \exp(q \cdot k/\tau)}, \tag{5}$$

where p is a positive sample and $k \in \mathbf{K}$ is a key sample. τ is a temperature parameter (set as 0.07 following [38]). Two distinct properties of InfoNCE when compared with the HE loss are: 1) only a single positive sample is considered, and 2) all key samples are used in the loss calculation.

Since this loss cannot be directly applied to our scenario where multiple positive examples exist, it should be modified to account for multi-label cases where multiple positive examples are allowed. [19] showed that multi-labeled

InfoNCE can be designed in two ways[2] to acquire this ability as:

$$\mathcal{L}_q^{i,in} = -\log \sum_{p \in \mathbf{P}_q} \left(\frac{\exp(q \cdot p/\tau)}{\sum_{k \in \mathbf{K}} \exp(q \cdot k/\tau)} \right), \tag{6}$$

$$\mathcal{L}_q^{i,out} = -\sum_{p \in \mathbf{P}_q} \left(\log \frac{\exp(q \cdot p/\tau)}{\sum_{k \in \mathbf{K}} \exp(q \cdot k/\tau)} \right), \tag{7}$$

where $\mathbf{P}_q$ is the set of positive samples for the query q.

On top of the vanilla InfoNCE, we have generated several variants by adjusting one or both of the "two distinct properties" as shown in the table below:

method	$C \times N$	w/ hard mining	mAP
InfoNCE (eq. 5)	128×2	no	62.9
InfoNCE (eq. 5)	128×2	yes	63.4
HE	128×2		77.5
InfoNCE (eq. 6)	16×16	no	70.5
InfoNCE (eq. 7)	16×16	no	72.8
InfoNCE (eq. 6)	16×16	yes	70.9
InfoNCE (eq. 7)	16×16	yes	73.1
HE	16×16		83.3

First of all, instead of using only a single positive sample (N=2), we can adjust the model to take more than one positive samples (N=16). Another form of variation is to feed InfoNCE with a selected number of key samples instead of using all of them for loss calculation. This variation is labeled as "w/ hard mining" where the number of hard mined samples are set to be equivalent to the case of HE loss (i.e., 15). Note that the InfoNCE with no variation at all (i.e., vanilla InfoNCE) is listed as the first in the table. Results can be interpreted in three aspects: i) using hard negative samples selected by hard negative mining was effective in InfoNCE loss, ii) multi-label variant was also effective compared to the single-label one, and iii) all the variants of InfoNCE loss still underperformed HE loss by a significant margin.

5 Discussion and Conclusion

We have achieved the state-of-the-art accuracy on all three re-ID benchmarks by adopting the MoReID architecture and the HE loss. Wrapping up the experiments and analyses we have included in the paper, it is worthwhile to leave several discussion points. First, we came across an interesting phenomenon in training where the loss values appeared as spikes, especially when the learning rates are high. Further study that takes into account other training parameters (e.g., different optimizer, various batch sizes) might be helpful to unravel the phenomenon. Second, although the usage of InfoNCE loss has recently exploded in self-supervised learning, we have observed that it did not live up to its fame when it was used for supervised re-ID. It will be worthwhile to evaluate how HE loss will fit into the self-supervised learning paradigm.

[2] In fact, most methods that use InfoNCE which is modified for supervised learning use one of these two. [25] used $\mathcal{L}_q^{i,in}$ while [10, 45, 21] used $\mathcal{L}_q^{i,out}$.

References

1. Aich, A., Zheng, M., Karanam, S., Chen, T., Roy-Chowdhury, A.K., Wu, Z.: Spatio-temporal representation factorization for video-based person re-identification. In: ICCV (2021)
2. Chen, H., Lagadec, B., Bremond, F.: ICE: Inter-instance contrastive encoding for unsupervised person re-identification. In: ICCV (2021)
3. Chen, P., Liu, W., Dai, P., Liu, J.: Occlude them all: Occlusion-aware attention network for occluded person Re-ID. In: ICCV (2021)
4. Chen, T., Ding, S., Xie, J., Yuan, Y., Chen, W., Yang, Y., Ren, Z., Wang, Z.: ABD-Net: Attentive but diverse person re-identification. In: ICCV (2019)
5. Chen, X., Xie, S., He, K.: An empirical study of training self-supervised vision transformers. In: ICCV (2021)
6. Cubuk, E.D., Zoph, B., Mane, D., Vasudevan, V., Le, Q.V.: AutoAugment: Learning augmentation policies from data. In: CVPR (2019)
7. Deng, J., Guo, J., Xue, N., Zafeiriou, S.: ArcFace: Additive angular margin loss for deep face recognition. In: CVPR (2019)
8. Fu, D., Chen, D., Bao, J., Yang, H., Yuan, L., Zhang, L., Li, H., Chen, D.: Unsupervised pre-training for person re-identification. In: CVPR (2021)
9. Goyal, P., Dollár, P., Girshick, R., Noordhuis, P., Wesolowski, L., Kyrola, A., Tulloch, A., Jia, Y., He, K.: Accurate, large minibatch SGD: Training ImageNet in 1 hour. arXiv:1706.02677 (2018), https://arxiv.org/abs/1706.02677
10. Gunel, B., Du, J., Conneau, A., Stoyanov, V.: Supervised contrastive learning for pre-trained language model fine-tuning. In: ICLR (2021)
11. Hao, X., Zhao, S., Ye, M., Shen, J.: Cross-modality person re-identification via modality confusion and center aggregation. In: ICCV (2021)
12. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: CVPR (2020)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
14. He, L., Liao, X., Liu, W., Liu, X., Cheng, P., Mei, T.: FastReID: A pytorch toolbox for general instance re-identification. arXiv:2006.02631 (2020), https://arxiv.org/abs/2006.02631
15. Hermans, A., Beyer, L., Leibe, B.: In defense of the triplet loss for person re-identification. arXiv:1703.07737 (2017), https://arxiv.org/1703.07737
16. Hoffer, E., Ailon, N.: Deep metric learning using triplet network. In: International Workshop on Similarity-Based Pattern Recognition (2015)
17. Huang, Y., Wu, Q., Xu, J., Zhong, Y., Zhang, Z.: Clothing status awareness for long-term person re-identification. In: ICCV (2021)
18. Khorramshahi, P., Peri, N., cheng Chen, J., Chellappa, R.: The devil is in the details: Self-supervised attention for vehicle re-identification. In: ECCV (2020)
19. Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C.: Supervised contrastive learning. In: NeurIPS (2020)
20. Kumar, R., Weill, E., Aghdasi, F., Sriram, P.: Vehicle re-identification: an efficient baseline using triplet embedding. In: IJCNN (2019)
21. Lee, H., Kwon, H.: Self-supervised contrastive learning for cross-domain hyperspectral image representation. In: ICASSP (2022)
22. Li, M., Huang, X., Zhang, Z.: Self-supervised geometric features discovery via interpretable attention for vehicle re-identification and beyond. In: ICCV (2021)

23. Li, Y., He, J., Zhang, T., Liu, X., Zhang, Y., Wu, F.: Diverse part discovery: Occluded person re-identification with part-aware transformer. In: CVPR (2021)
24. Meng, D., Li, L., Liu, X., Li, Y., Yang, S., Zha, Z.J., Gao, X., Wang, S., Huang, Q.: Parsing-based view-aware embedding network for vehicle re-identification. In: CVPR (2020)
25. Miech, A., Alayrac, J.B., Smaira, L., Laptev, I., Sivic, J., Zisserman, A.: End-to-end learning of visual representations from uncurated instructional videos. In: CVPR (2020)
26. van den Oord, A., Li, Y., Vinyals, O.: Representation learning with contrastive predictive coding. arXiv:1807.03748v2 (2018), https://arxiv.org/abs/1807.03748v2
27. Park, H., Lee, S., Lee, J., Ham, B.: Learning by aligning: Visible-infrared person re-identification using cross-modal correspondences. In: ICCV (2021)
28. Rao, Y., Chen, G., Lu, J., Zhou, J.: Counterfactual attention learning for fine-grained visual categorization and re-identification. In: ICCV (2021)
29. Schroff, F., Kalenichenko, D., Philbin, J.: FaceNet: A unified embedding for face recognition and clustering. In: CVPR (2015)
30. Sohn, K.: Improved deep metric learning with multi-class n-pair loss objective. In: NeurIPS (2016)
31. Sun, Y., Cheng, C., Zhang, Y., Zhang, C., Zheng, L., Wang, Z., Wei, Y.: Circle loss: A unified perspective of pair similarity optimization. In: CVPR (2020)
32. Ulyanov, D., Vedaldi, A., Lempitsky, V.: Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In: CVPR (2017)
33. Wang, F., Cheng, J., Liu, W., Liu, H.: Additive margin softmax for face verification. IEEE Sign. Process. Letters **25**(7), 926–930 (Jul 2018)
34. Wang, F., Xiang, X., Cheng, J., Yuille, A.L.: Normface: L_2 hypersphere embedding for face verification. In: ACM MM (2017)
35. Wang, H., Wang, Y., Zhou, Z., Ji, X., Gong, D., Zhou, J., Li, Z., Liu, W.: CosFace: Large margin cosine loss for deep face recognition. In: CVPR (2018)
36. Wang, X., Girshick, R., Gupta, A., He, K.: Non-local neural networks. In: CVPR (2018)
37. Wang, X., Hua, Y., Kodirov, E., Robertson, N.M.: Ranked list loss for deep metric learning. In: CVPR (2019)
38. Wu, Z., Xiong, Y., Yu, S., Lin, D.: Unsupervised feature learning via non-parametric instance discrimination. In: CVPR (2018)
39. Yan, C., Pang, G., Jiao, J., Bai, X., Feng, X., Shen, C.: Occluded person re-identification with single-scale global representations. In: ICCV (2021)
40. Yan, C., Pang, G., Wang, L., Jiao, J., Feng, X., Shen, C., Li, J.: BV-Person: A large-scale dataset for bird-view person re-identification. In: ICCV (2021)
41. Zhang, L., Rusinkiewicz, S.: Learning local descriptors with a CDF-based dynamic soft margin. In: ICCV (2019)
42. Zhang, X., Zhang, R., Cao, J., Gong, D., You, M., Shen, C.: Part-guided attention learning for vehicle instance retrieval. IEEE Trans. Intell. Transp. Syst. (2020)
43. Zhao, J., Zhao, Y., Li, J., Yan, K., Tian, Y.: Heterogeneous relational complement for vehicle re-identification. In: ICCV (2021)
44. Zheng, K., Liu, W., He, L., Mei, T., Luo, J., Zha, Z.J.: Group-aware label transfer for domain adaptive person re-identification. In: CVPR (2021)
45. Zheng, M., Wang, F., You, S., Qian, C., Zhang, C., Wang, X., Xu, C.: Weakly supervised contrastive learning. In: ICCV (2021)

46. Zheng, Y., Tang, S., Teng, G., Ge, Y., Liu, K., Qin, J., Qi, D., Chen†, D.: Online pseudo label generation by hierarchical cluster dynamics for adaptive person re-identification. In: ICCV (2021)
47. Zhong, Z., Zheng, L., Kang, G., Li, S., Yang, Y.: Random erasing data augmentation. In: AAAI (2020)