# Swap Path Network for Robust Person Search Pre-training

## Supplementary Material

*This supplementary material contains content beneficial to but not required for understanding of the main paper. It includes information about code used to produce results from the paper, additional dataset metadata, implementation details, and results of additional ablations which explore the hyperparameter design space.*

## 6. Code and Reproduction

We include all code, configs, and instructions required to reproduce results from the paper in the corresponding GitHub repository[2].

## 7. Dataset Metadata

The two standard person search benchmark datasets we use for model fine-tuning are the CUHK-SYSU dataset [46] and the PRW dataset [52].

The CUHK-SYSU dataset has 18k scenes with 95k annotations, split among 23k boxes for 8.5k known identities, and 72k boxes for unknown identities. The scenes in CUHK-SYSU come from a mixture of handheld device photos taken on city streets, and scenes from TV shows and films. The standard test retrieval scenario for CUHK-SYSU uses 2,900 queries with 100 scenes in each gallery.

The PRW dataset has 12k scenes with 43k annotations, split among 34k boxes for 1k known identities, and 9k boxes for unknown identities. The scenes from PRW come from six fixed cameras installed at the Tsinghua University campus in Beijing. The standard test retrieval scenario for PRW uses 2,057 queries with all 6,112 test scenes in the gallery.

## 8. Additional Implementation Details

**Model Configurations.** In all model versions, a 4-layer MLP is applied to offset embeddings $x_o$ to produce box regression offsets, while classification logits are computed directly from $\|x_o\|$. The training batch size is 8, with $k = 2$ augmentations per image during pre-training.

**Anchor Sampling.** To reduce differences between QC and OC pre-training from anchor sampling during detection, we ensure that the same number of anchors are optimized in each batch between QC and OC trials (2,048 by default). For the OC case, this is calculated as number of images per batch $\times$ number of anchors per image. For the QC case, this is calculated as number of images per batch $\times$ number of query-image pairs $\times$ number of anchors per query-image pair.

Query-image pairs constitute the pairing of a given query embedding $x_q$ with anchor embeddings from a given image $x_a$. With the total number of query-image pairs equal to number of queries $\times$ number of images, this can quickly exceed memory limitations if all pairs are used, so some subsampling procedure is required. We select pairs in the following order up to a fixed number (32 by default): 1) queries are selected for images which they are not present in, but share a label with a box in the image, 2) queries are selected for images which they are present in, and 3) queries are selected for images which they are neither present in, nor share a label with any boxes in.

Both QC and OC sampling methods attempt to balance the number of positive and negative samples, but there are usually more negative samples in practice due to sparsity of positives and typical hyperparameter selection.

**Image Augmentation.** We adopt three methods of image augmentation, which all consist of combinations of scaling, cropping, and horizontal flipping. For consistency, we label these here using the configuration name used in the code. We call the standard augmentation for person search *window resize* as in [23], abbreviated `wrs`. This consists of scaling the image to fit in a window with shortest side length 900 and longest side length $1,500$. `wrs` augmentation is used only for evaluation. In `rrc2` augmentation, we first perform `wrs` scaling, then randomly select from two types of crops (followed by random horizontal flip): 1) fixed size square random crop containing at least one bounding box in the image, chosen at random, and 2) random sized crop containing all bounding boxes in the image, that is then resized to a fixed square size. These square crop sizes are $512 \times 512$ during pre-training and SPNet-S fine-tuning, and $1024 \times 1024$ during SPNet-L fine-tuning. Crop size is one of the most important parameters for controlling memory usage. `rrc2` augmentation is used for all fine-tuning runs. `rrc_scale` augmentation is the same as `rrc2` augmentation, except we scale randomly in the range $0.5\times$ to $2\times$ instead of `wrs` scaling. `rrc_scale` augmentation is used for all pre-training runs.

[2]Project repository: https://github.com/LLNL/spnet

| Model | Backbone | Pre-train (QC) COCOPersons | Pre-train (OC) COCOPersons | Fine-tune (OC) CUHK-SYSU | Fine-tune (OC) PRW |
|---|---|---|---|---|---|
| SPNet-S | ConvNeXt-T | 30.0h | 24.5h | 2.5h | 1.5h |
| SPNet-L | ConvNeXt-B | 46.5h | 40.5h | 9.0h | 4.5h |

Table 4. Training times for person search model variants on a single A100 GPU. Pre-training and fine-tuning are each done for 30 epochs on the respective datasets.

| | CUHK-SYSU | | | PRW | | |
|---|---|---|---|---|---|---|
| **Pre-train Method** | mAP | GT mAP | AP@0.5 | mAP | GT mAP | AP@0.5 |
| Random Init. | 69.8 | 85.8 | 47.8 | 22.3 | 28.7 | 66.8 |
| Classifier | 91.9 | 95.8 | 83.2 | 52.7 | 56.6 | 88.8 |
| Ours-OC | 92.8 | **96.3** | 83.9 | 54.5 | 57.6 | **89.1** |
| Ours-QC | **93.3** | 95.9 | **84.9** | **55.6** | **58.7** | 88.9 |

Table 5. Person search (mAP), re-id (GT mAP), and detection metrics (AP@0.5) for pre-training method comparison on SPNet-S with ConvNeXt-Tiny backbone.

**Training Times.** All models were trained using a single A100 GPU with 82GB VRAM. Pre-training and fine-tuning times are shown in Tab. 4, with the QC pre-training taking 30 hours for SPNet-S and 46.5 hours for SPNet-L. We note that OC pre-training is more efficient, taking about 6 hours less for either model variant.

# 9. Supporting Ablations

## 9.1. QC vs. OC Ablations

**Subtask Performance.** In Tab. 5, we compare QC vs. OC pre-training to ImageNet-1k Classifier pre-training, and random backbone initialization. The comparison is done for SPNet-S with a ConvNeXt-Tiny backbone, and we show metrics of person search (mAP) in addition to metrics of re-id (GT mAP) and detection (AP@0.5). This helps us understand the contribution of pre-training to detection, re-id, and the combined person search problem. Measuring ground truth re-id performance (GT mAP) is equivalent to using a perfect object detector, and helps us understand re-id performance independent of detector quality.

We show both QC and OC pre-training improve over ImageNet-1k Classifier pre-training for all metrics, and that QC is superior to OC pre-training for most metrics. Most importantly, we show that QC pre-training exceeds OC pre-training for person search (mAP) on both datasets (+0.5% on CUHK-SYSU, +1.1% on PRW), though whether re-id or detection benefit more depends on the dataset. Finally, we show that all pre-training vastly exceeds random initialization. Additional ablations comparing QC vs. OC pre-training and fine-tuning are shown in the following subsections. These results show that QC pre-training outperforms OC pre-training across a range of different conditions.

**Re-id Pre-training.** In Tab. 6, we compare different methods of handling the re-id loss during pre-training. The goal was to isolate the effect of the re-id loss on pre-training, for both QC and OC methods. We found that pre-training only the re-id loss, without optimizing the other losses, was not beneficial for all statistics.

We found that QC pre-training using detected boxes with IoU $\geq 0.7$ performed best on balance. When only ground truth (GT) boxes were used to compute the re-id loss, and no detected boxes, we found that QC pre-training was impaired more than OC pre-training.

**Query-Centric vs. Object-Centric Pre-training.** We compare query-centric vs. object-centric pre-training for four configurations, given in Tab. 7 (bottom), with results in Fig. 7 for the PRW dataset. All models with pre-training significantly outperform the baseline in mAP and top-1 accuracy, and nearly all QC models outperform the corresponding OC model. One exception is the Frozen Backbone model for top-1 accuracy, showing that the query-centric pre-training performs better when the backbone is optimized in addition to later layers.

For pre-training, we also explore the effect of the *Learning without Forgetting* (LwF) loss from [26], equivalent to the knowledge distillation loss from [20]. Since we initialize our model backbone from ImageNet-1k classifier weights before pre-training, the idea is that this loss will help preserve useful features learned during the classifier pre-training. This loss

| Method | CUHK-SYSU mAP | CUHK-SYSU top-1 | PRW mAP | PRW top-1 |
|---|---|---|---|---|
| Baseline | 91.9 | 93.2 | 52.7 | 86.6 |
| Re-id Only | 91.5 | 93.0 | 51.9 | 87.4 |
| OC Re-id GT | 92.4 | 93.7 | **55.7** | 89.1 |
| QC Re-id GT | 93.2 | **94.3** | 55.1 | 88.5 |
| OC Original | 92.8 | 94.1 | 54.5 | 88.8 |
| QC Original | **93.3** | 94.1 | 55.6 | **89.5** |

Table 6. Comparison of different settings of the re-id pre-training loss for fine-tuning performance. For *re-id GT*, only GT box embeddings are used to compute re-id loss, and for *re-id only*, we optimize only the re-id loss (with GT boxes) and no other losses. In original trials, both GT and detected box embeddings are used to compute re-id loss.

| Config. Name | Ablation Name | Cascade Steps | Shared Heads | Re-id Dim |
|---|---|---|---|---|
| SPNet-S | c0-share-d128 | 0 | ✓ | 128 |
| - | c0-sep-d128 | 0 | | 128 |
| - | c2-share-d128 | 2 | ✓ | 128 |
| - | c2-sep-d128 | 2 | | 128 |
| SPNet-L | c2-sep-d2048 | 2 | | 2048 |

| | | Backbone | | Post-Backbone | | |
|---|---|---|---|---|---|---|
| Config. Name | Ablation Name | LR | WD | LR | WD | LwF Loss |
| - | Uniform LR | 1e−4 | 1e−3 | 1e−4 | 1e−3 | |
| Pre-train | Mixed LR | 1e−5 | 0 | 1e−4 | 1e−3 | |
| - | Mixed LR + LwF | 1e−5 | 0 | 1e−4 | 1e−3 | ✓ |
| - | Frozen-BB | Frozen | Frozen | 1e−4 | 1e−3 | |
| Fine-tune | Fine-Tune | 1e−4 | 5e−4 | 1e−4 | 5e−4 | |

Table 7. SPNet architecture (top) and layer optimization (bottom) hyperparameter configurations. LR = Learning Rate, WD = Weight Decay, LwF = Learning without Forgetting [26].

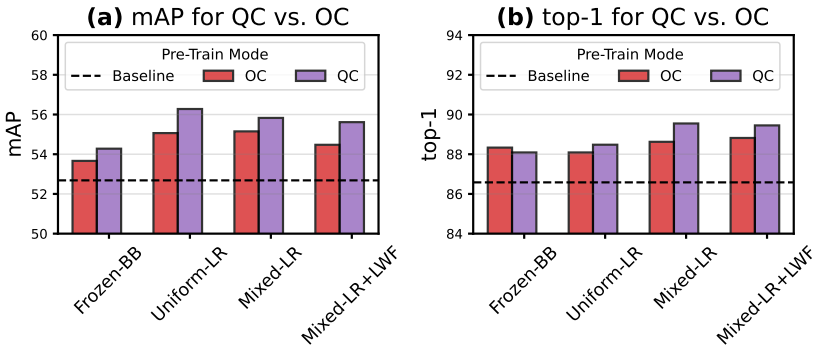


Figure 7. Comparison of SPNet fine-tune performance on PRW with QC vs. OC pre-training for varying layer optimization hyperparameters (configurations described in Tab. 7).

$\mathcal{L}_{\text{LwF}}$ is simply added to the other losses when used, using temperature $T = 2$ as in [26]. As shown in Fig. 7, we did not find use of the LwF loss to result in consistently better performance, so it was not used for other trials.

**Query-Centric vs. Object-Centric Fine-tuning.** While we primarily explore OC fine-tuning (OC-FT) and evaluation in this work, our framework supports QC fine-tuning (QC-FT) and evaluation as well. QC fine-tuning is done using the same

| Method | CUHK-SYSU mAP | CUHK-SYSU top-1 | PRW mAP | PRW top-1 |
|---|---|---|---|---|
| OC-FT | 91.9 | 93.2 | 52.7 | 86.6 |
| OC-FT+PT | 93.3 | 94.1 | 55.6 | 89.5 |
| *Gain from PT* | +1.4 | +0.9 | +2.9 | +2.9 |
| QC-FT | 80.8 | 87.7 | 54.0 | 86.7 |
| QC-FT+PT | 77.4 | 85.4 | 54.1 | 87.1 |
| *Gain from PT* | -3.4 | -2.3 | +0.1 | +0.4 |

Table 8. Comparison of QC and OC fine-tuning (-FT) for baseline model vs. COCOPersons QC pre-trained model (+PT).

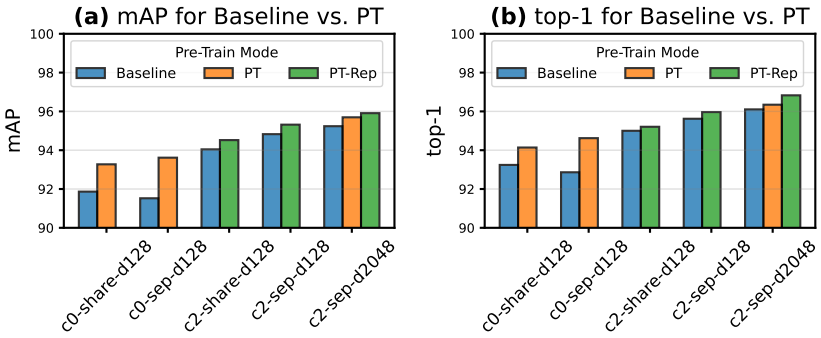


Figure 8. Comparison of SPNet fine-tune performance on CUHK-SYSU with and without QC pre-training for varying architecture hyper-parameters (configurations described in Tab. 7). PT=pre-trained, PT-Rep=pre-trained with replicated weight loading for cascade layers.

procedure as QC pre-training, with $k = 2$ augmentations per image, but using the `rrc2` augmentation method.

In Tab. 8, we compare QC-FT and OC-FT for models with and without COCOPersons QC pre-training. While the QC-FT model outperforms the OC-FT model on PRW top-1 (without PT), OC-FT performance is drastically higher for mAP and top-1 on CUHK-SYSU (>10% mAP). In addition, the OC-FT model benefits significantly from QC pre-training for both datasets and metrics, while the QC-FT model benefits only slightly on PRW and is actually harmed on CUHK-SYSU.

This suggests QC fine-tuning is more impacted by the distribution shift between pre-training and fine-tuning datasets, at least for the shared embedding head model. It also suggests the QC pre-train to OC fine-tune transition has a regularizing effect, limiting the effect of overfitting on the pre-training dataset.

## 9.2. Model Architecture Ablation

**Model Architecture.** To understand the model architecture design space, we examine variations in number of cascade steps, shared vs. separate embeddings heads, and size of the re-id embedding dimension. In Fig. 8, we compare the configurations described in Tab. 7 (top). We show that the cascaded model with two steps is better than the base model, using separate embedding and re-id heads results in better fine-tuning performance, and larger re-id embedding dimension is beneficial. In addition, all architecture configurations benefit from pre-training, although the cascaded models benefit less.

**Cascaded Weight Loading.** When loading weights from pre-trained SPNet into a larger version with multiple cascade stages, we have to make a choice about how to load weights into layers not utilized during pre-training. We explore two simple options: loading weights only into layers that were pre-trained, and duplicating weights into equivalent layers in each cascade stage, shown by *PT* vs. *PT-Rep* in Fig. 8. For the `c2-sep-d2048` configuration, the PT-Rep method of weight loading outperforms PT, showing that the benefits of the pre-trained weight initialization extend to cascade stages, even though the pre-trained model was trained without cascading.

**Classifier Logits.** Recall from Sec. 3.3 that the offset embedding is defined as the difference between query and anchor embeddings: $x_o = x_q - x_a$ with $x_o \in \mathbb{R}^d$ and $w = \|x_o\|$. Then the class logits $z$ can be calculated as

| Method | CUHK-SYSU | | PRW | |
|---|---|---|---|---|
| | mAP | top-1 | mAP | top-1 |
| BatchNorm Logits (+) | 90.4 | 91.9 | 49.8 | 84.3 |
| BatchNorm Logits (-) | 91.2 | 92.0 | 42.2 | 82.2 |
| FixedNorm Logits (+) | 91.1 | 92.2 | 52.6 | 86.3 |
| FixedNorm Logits (-) | **91.9** | **93.2** | **52.7** | **86.6** |

Table 9. Comparison of BatchNorm Logits from [6] to proposed FixedNorm Logits for CUHK-SYSU and PRW datasets, for the baseline model. (+) vs. (-) indicates whether positive or negative embedding norm was used to compute logits.

$$z = \frac{w - \mu_{\text{Chi}}}{s_{\text{Chi}}} \tag{3}$$

with

$$\mu = \sqrt{2}\frac{\Gamma\left(\frac{d+1}{2}\right)}{\Gamma\left(\frac{d}{2}\right)}, \quad s = \sqrt{d - \mu^2} \tag{4}$$

We note for $d \gg 1$, $\mu \approx \sqrt{d - \frac{1}{2}}$ and $s \approx \sqrt{\frac{1}{2}}$.

The *FixedNorm* transformation in Eq. (3) has two key properties: 1) For $x_o \sim \mathcal{N}(0, I_d)$, $\mathbb{E}[z] = 0, \text{Var}[z] = 1$, independent of $d$. In addition, $\lim_{d\to\infty} z \sim \mathcal{N}(0, 1)$ due to the Central Limit Theorem. Although the distribution of $x_o$ is rarely unit-normal and changes during optimization, this framing gives us a reasonable choice of shifting and scaling parameters which works well in practice, does not require learnable parameters or hyperparameters, and holds independent of $d$.

2) For $x_q$ similar to $x_a$ *i.e.*, $\|x_o\|$ small, $z$ is larger, and for $x_q$ dissimilar to $x_a$ *i.e.*, $\|x_o\|$ large, $z$ is smaller. Intuitively, when a query embedding matches a given anchor embedding, according to box IoU overlap, we want the two to be more similar, and when they do not match, we want them to be more different. The relationship is also critical to co-optimizing with the re-id loss on $x_q$ and has a nice relationship with the box regression loss.

To validate our FixedNorm logits over the BatchNorm logits from [6], we compare both for the baseline model, with results shown in Tab. 9. The FixedNorm method achieves greater performance for all metrics, especially on the PRW dataset, where mAP exceeds the BatchNorm method by more than 10%. This is likely due in part to unbalanced batch statistics, which are dominated by negative samples, unlike the original Norm-Aware Embedding use case. Tab. 9 also shows that negating the norm after scaling is beneficial, especially for the FixedNorm logits, validating the rationale discussed above.

## Acknowledgements

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344, with release number LLNL-CONF-2001396.