Continuous Learning without Forgetting for Person Re-Identification

Continuous Learning without Forgetting for Person Re-Identification

Nehemia Sugianto*, Dian Tjondronegoro+, Golam Sorwar*,
Prithwi Chakraborty*, Elizabeth Irenne Yuwono*
Southern Cross University*, Griffith University+
Queensland, Australia
{n.sugianto.11,e.yuwono.10}@student.scu.edu.au, d.tjondronegoro@griffith.edu.au,
{golam.sorwar,prithwi.chakraborty}@scu.edu.au

Abstract

Deep learning-based person re-identification faces a scalability challenge when the target domain requires continuous learning. Service environments, such as airports, need to recognize new visitors and add new cameras over time. Training-at-once is not enough to make the model robust to new tasks and domain variations. A well-known approach is fine-tuning, which suffers forgetting problem on old tasks when learning new tasks. Joint-training can alleviate the problem but requires old datasets, which is unobtainable in some cases. Recently, Learning without forgetting (LwF) shows its ability to mitigate the problem without old datasets. This paper extends the benefit of LwF from image classification to person re-identification with further challenges. Comprehensive experiments are based on Market1501 and DukeMTMC4ReID to evaluate and benchmark LwF to other approaches. The results confirm that LwF outperforms fine-tuning in preserving old knowledge and joint-training in faster training.

1. Introduction

Service environments often use multiple cameras to monitor crowd behavior, such as for analyzing airport passenger flow. Non-overlapping cameras can track people as they move from the entrance to the next sequence of their journey, such as check-in, security, and departure. Automatic tracking requires person re-identification models, which use a database of person-of-interest images, known as the gallery, to differentiate people's appearance visually. The size and complexity of the gallery need to increase over time to make the models robust against new data and variations, hence requiring continuous training [1].

Existing continuous training approaches, such as feature extraction [2], fine-tuning [3], and joint-training [4] are not adequate to fully address catastrophic forgetting problems [5], known for forgetting gained knowledge of old tasks while learning new tasks. By incorporating learning using both old and new data, joint-training can theoretically solve the catastrophic forgetting. However, for practicality, old data is not always available in some cases, and processing an accumulated amount of data requires more substantial computation time and storage. A recent approach called learning without forgetting (LwF) [6] aims to alleviate the limitations of joint-training by supporting learning of old tasks without access to the original datasets. Consequently, it offers comparable performance for both old and new tasks with faster training time and cheaper computation.

This paper extends LwF from its original application of image classification to person re-identification tasks. Comprehensive experiments are conducted on various training methods and training strategies on Market1501 [7] and DukeMTMC4Re-ID [8] to demonstrate LwF's benefits compared to the existing training approaches.

2. Related works

In a recent survey, existing deep learning-based person re-identification methods achieve encouraging results on large-scale person re-identification datasets but degrade when applied in real-world environments that require robustness against many cameras and variations across a long period [9]. The training-at-once approach relies on short-term-collected datasets, which over time would struggle to achieve and maintain sufficient generalization for dealing with rapidly growing new data. Even the most recent large-scale dataset for person re-identification [8] only has approximately 46,261 pedestrian boxes over 1,852 identities, which is much smaller compared to the state-of-the-art face recognition datasets, such as the Caltech dataset [10] with 350,000 pedestrian boxes over 2,300 identities.

For continuous training with new data, **joint-training** [4] is a common technique to train a network from scratch as new tasks are added. It can achieve good performance in both old and new tasks. However, it may not be viable when old datasets may be unobtainable in some cases, and processing all images require expensive computation. Consequently, joint-training suffers from longer training

978-1-5386-9294-3/18/$31.00 ©2019 IEEE

time and substantial memory usage as more images are fed into the network to update the weights. This problem may get even worse when the problem domain requires a seemingly never-ending stack of new images over time, which raise issues with storage size and memory capacity.

Transfer learning approaches have been proposed to solve the issue with re-processing old data. **Fine-tuning** [3] is a popular technique to learn new tasks by adapting existing knowledge for new tasks or domains. It can improve the performance significantly when dataset of new tasks is limited. For example, adapting a network previously trained for image classification to facial expression recognition and person re-identification tasks [11]. Using this technique, knowledge of features that are discriminative for identities, attributes and semantic can transferred without any supervised learning for a new target domain [12]. For instance, GoogleNet can provide the base knowledge for a multilevel triplet deep learning model to compute multilevel features efficiently for person re-identification tasks [13]. Similarly, models which use global features can benefit from the transferred knowledge for mapping local features to person re-identification tasks [14]. While fine-tuning works well on new tasks, it suffers forgetting problem on old tasks due to the absence of updating the weights for the original datasets. Arguably, the nature of deep learning model architecture is the main cause. Internal representation of hidden layers are overlapping and any small change in a single neuron can affect multiple representation at the same time [15]. When backpropagation is performed to update weights, all parameters will be affected as they are all connected to each other [16].

Knowledge distillation, also known as domain adaptation, is another technique for transferring knowledge from one network to another for the same tasks. It was originally proposed by [17] to transfer knowledge from larger network to smaller network for more efficient deployment. Knowledge distillation can prevent catastrophic forgetting problem, as during training, distillation uses a loss function to maintain the performance of old tasks while learning new tasks. The main idea is to encourage large (old) network and small (new) network have similar responses as close as possible by using a modified cross-entropy loss. Knowledge can be transferred by applying an extra guided middle layer to capture the intermediate representation, in order to build a thin but deeper student network [18]. Distillation is also used to generate a deeper and wider network which has equivalent capabilities to an existing one [19]. This method can quickly initialize the new network, even with a different structure, through faster hyper-parameter exploration that approximates the original network.

Learning without forgetting (LwF) [6] is a recent technique built on knowledge distillation to learn new tasks while maintaining the performance on old tasks, without requiring or processing the original dataset. The main idea is to use a modified loss function consisting of cross-entropy loss for learning new tasks combined with a distillation loss function for maintaining old tasks. Unlike the original knowledge distillation, which requires the original training data, this technique instead distills the new tasks' images to predict the similar responses for old tasks. Based on the experiment results, LwF can bridge the gaps between the pros and cons of fine-tuning and joint-training. Compared to fine-tuning, LwF's performance for new tasks is still lower, but it is better at retaining the performance of the old tasks with only slightly more computation. Compared to joint-training, LwF's performance for both old and new tasks is still lower, but can be trained faster without reprocessing the old datasets.

Another similar work is proposed in [20], which uses distillation for lifelong learning via progressive distillation and retrospection for image classification tasks. However, it requires a large number of samples for each class, as the training process needs to use an updated set of training images for retrospection purposes. Similarly, iCaRL method [21] is introduced to maintain both old and new tasks' performance for image classification while new classes are added incrementally. This technique has been extended for generic object detection tasks [22].

As most recent works in person re-identification have been primarily based on a training-at-once approach, this paper proposes the benefits of using LwF to bridge the gaps between fine-tuning and joint-training. This paper extends the original scope of LwF and addresses further challenges in person re-identification tasks. The evaluation is based on a comprehensive experiment using widely used and publicly available datasets.

3. Method

The proposed person re-identification adopts ResNet as the base model, which was trained for image classification on ImageNet. The key contribution of this work is to evaluate the benefits and disadvantages of applying different training techniques for incremental learning.

3.1. ResNet base model

The base model is used for demonstrating transfer learning, adapting from object detection to person re-identification tasks. It has two parts: feature extractor is used to extract visual features of an image fed into the network; and classification is used to perform classification during training. Cosine similarity is used to calculate the distance between two images during testing.

Feature extractor: ResNet [23] convolutional layers are used for extracting features. Compared to AlexNet, VGG16 and other existing networks, ResNet can stack residual components to pass more information into the subsequent layers to extract discriminative features for

person re-identification effectively. As the model gets deeper, with more stacks of residual layers, the performance increases and is able to handle a large number of people in the gallery. ResNet50 is chosen among other ResNet variants, as it can achieve a comparable accuracy with fewer parameters. It has five blocks which enables us to extract discriminative features of a person based on visual features.

Classification: Fully connected layers are placed at the last part of the network. The first layer has 1,000 inputs representing the number of output nodes of the last layer of feature extractor. Next is a fully connected layer, which has a number of output nodes that corresponds to the number of people that needs to be recognized. Classification part is only used during training, as during testing of the model, an image is fed into the network up to the feature extraction part only. A cosine distance algorithm uses the extracted features to find the most similar images. Scalability is the motivation for using this approach, instead of classification, since the number of people recognized may become very large in future and it would seem impossible to be accommodated in the classification layer. Moreover, continuously increasing the number of neurons and parameters to handle more-and-more people will eventually become prohibitive. Using similarity-based person re-identification will also enable us to search completely new people as the model is robust enough to differentiate the discriminative features.

Measuring similarity: verification-based person re-identification model [9] is applied to calculate similarities between input images. Initially, a query image is fed into the network to extract the features. Then, each feature of the query image is compared against the features of each image in the gallery set to calculate cosine similarity scores. The final output is the *top-10* most similar images that are closest to the query. The first-ranked image can be considered as the most probable matching person searched.

3.2. Continuous training techniques

This paper aims to benchmark which training approach works best in the target application. To mimic continuous learning, each network is trained incrementally in *N* stages. Therefore, each dataset is distributed into *N* sets of images equally. Training images are distributed to represent *people addition* (new person added over time) or *cameras addition* (more samples of a person from new cameras added over time). For each training stage, the new set of images is considered as new tasks and the previous sets of images are considered as old tasks. Once a model is retrained, the weights are stored and used for subsequent training. This strategy is to evaluate the network's performance for old and new tasks, as more people and cameras are added over time.

Definitions: i represents the current training stage number. θ_s is the shared convolutional neural networks (*CNN*) and fully connected (*FC*) layers; θ_o is the output layer and corresponding weights for an old task; and θ_n is the randomly initialized task-specific parameters for new tasks. θ_o and θ_n can be regarded as classifiers that operate on features parameterised by θ_s. X_o, X_n are the training data; while Y_o, Y_n are the ground truth data; and Y'_o, Y'_n are the output predictions for old and new tasks respectively. Notations are presented visually in Figure 2.

Figure 1 describes the algorithms for different training approaches used to train θ_n while benefitting from the previously learned θ_s in each training. Regardless of the training approach applied, the weights from previous training stage CNN_{i-1} are initially loaded when the model is already trained, followed by adding all *N* nodes for the new tasks at the last FC layer. Then, the new weights and biases are initialized. The number of new parameters is equal to *N* times the number of neurons in the previous FC layer. Adding new nodes at the last layer, instead of expanding the network further, can prevent the increasing number of parameters significantly. Figure 2 shows the difference between each training approach.

Initialization:
Determine if training method selected is fine-tuning (FT), joint-training (JT) or learning without forgetting (LwF)
If i >1:
  CNN_i ← Load weight of CNN_{i-1} + add N new nodes at FC layer
  θ_n ← Initialize the new weights and biases (θ_n)
If LwF:
  Y_o ← CNN_{i-1} (X_n, θ_s, θ_o) //to compute old data's ground truth

Training:
  If LwF:
    Calculate Y'_o: CNN_i (X_n, θ_s, θ_o)
  Else if JT:
    Calculate Y'_o: CNN_i (X_o, θ_s, θ_o)
  Calculate Y'_n: CNN_i (X_n, θ_s, θ_n)
  If LwF:
    Calculate loss: DistillationLoss (Y'_o, X_n) + CrossEntrophyLoss(Y'_n, X_n)
  Else if JT:
    Calculate loss: CrossEntrophyLoss(Y'_o, X_o) + CrossEntrophyLoss(Y'_n, X_n)
  Else if FT:
    Calculate loss: CrossEntrophyLoss(Y'_n, X_n)

Figure 1: Sequence of various training approaches

Joint-training (JT). Using new tasks' data X_n and its ground truth Y_n, fine tune θ_s and θ_n for the new tasks while at the same time fine tune θ_s and θ_o using old task's data X_o and its ground truth Y_o as well. All shared parameters θ_s, θ_o, θ_n are unfrozen. Cross entropy loss is used to calculate loss on both tasks and backpropagation is performed to optimize all parameters θ_s, θ_o, θ_n at the same time.

Fine-tuning (FT). Using new tasks' data X_n and its ground truth Y_n, fine tune θ_s and θ_n for the new tasks while leaving θ_o unchanged (frozen) due to the absence of old tasks' X_o and its ground truth Y_o. Only θ_s, θ_n parameters in CNN are unfrozen. Cross entropy loss is used to calculate loss on new tasks and backpropagation is performed to optimize θ_s, θ_n parameters.

Learning without forgetting (LwF). Using new tasks data X_n and its ground truth Y_n, fine tune θ_s and θ_n for the new tasks while at the same time fine-tuning the θ_s and θ_o for the old tasks by using the generated output response as old tasks' X_o ground truth Y_o. The generated output responses are sets of class probabilities which are initially computed on old network CNN_{i-1} before training the network using new tasks' data X_n. The outputs are recorded and will be used as ground truth of old tasks X_o during training. All shared parameters θ_s, θ_o, θ_n are unfrozen.

To optimize all parameters θ_s, θ_o, θ_n, a modified cross entropy loss is used to calculate the loss on new and old tasks. *For new tasks*: a cross entropy loss function is used to calculate the loss based on the output prediction of the network since our tasks are multi-class classification (as described in Equation 1). y' denotes the softmax output of the network and y denotes the ground truth. *For old tasks:* a modified cross entropy loss function (as described in Equation 2) is used to calculate loss between output prediction on the network and its ground truth which is generated at the beginning on the old network. Knowledge distillation loss [17] is used to calculate loss for each output class (as described in Equation 3). L denotes the number of classes. T is experimentally set between 0.5 to 2 to get the best results. *Final loss* is calculated by the sum of loss from old tasks and new tasks. Total loss is averaged loss over all training images in a training batch.

$$Loss_{new}(y_{new}, y'_{new}) = \sum_{i=1}^{L} -y_{new}(i) \,.\log y'_{new}(i) \quad (1)$$

$$Loss_{old}(y_{old}, y'_{old}) = \sum_{i=1}^{L} -y_{old}(i) \,.\log y'_{old}(i) \quad (2)$$

$$y_{old}(i) = \frac{(y_{old}{}^{(i)})^{1/T}}{\sum_j (y_{old}{}^{(j)})^{1/T}} \; ; \; y'_{old}(i) = \frac{(y'_{old}{}^{(i)})^{1/T}}{\sum_j (y'_{old}{}^{(j)})^{1/T}} \quad (3)$$

4. Experimental setting

This paper conducts two comprehensive experiments to test proposed method's effectiveness. First is testing the performance of knowledge transfer against people and camera addition. Second is investigating the benefits and disadvantages from adopting the different training approaches (fine-tuning, joint-training, LwF).

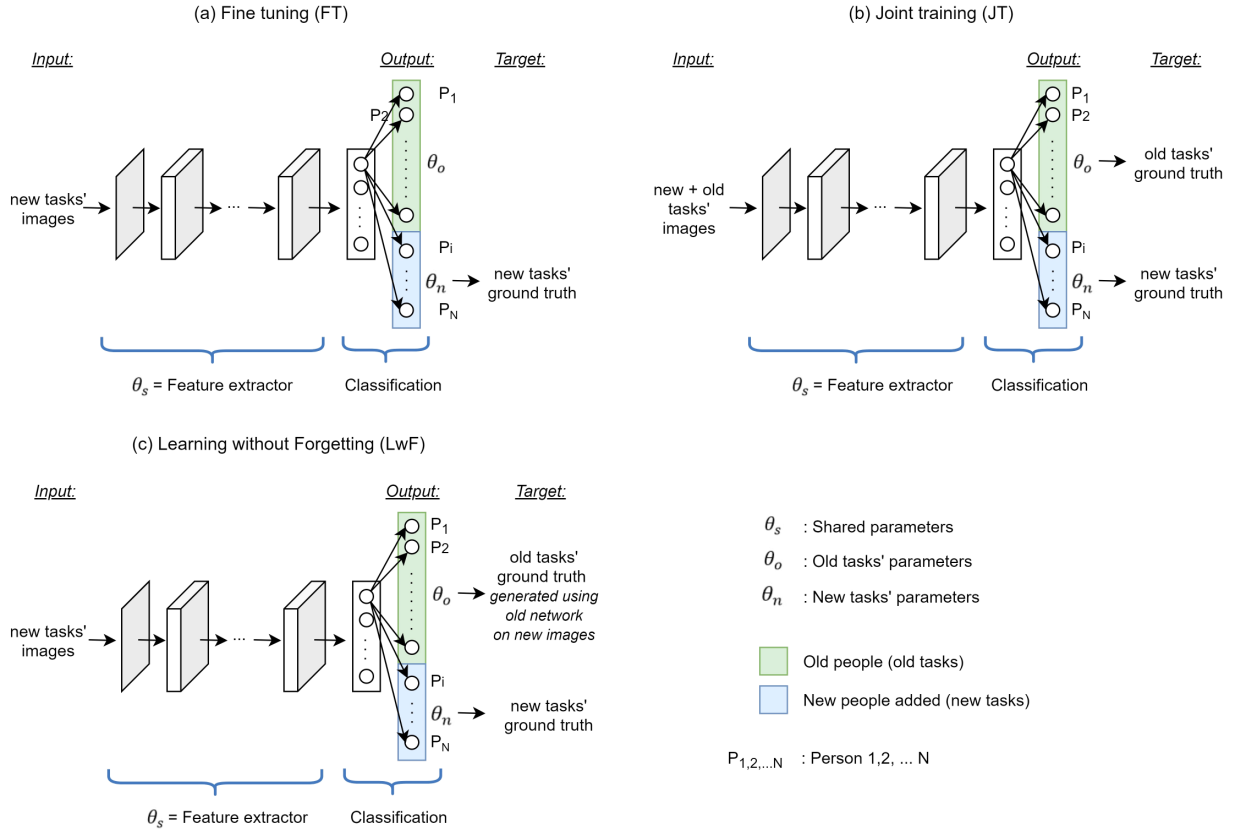

Figure 2: Three different training approaches used for the experiments (notations are described in Section 3.2).

The experiments are based on two large-scale datasets Market1501, DukeMTMC4ReID, which are publicly available and widely used for person re-identification representing pose, occlusion, and camera angle variations.

Definitions: *Training strategies* are 2 strategies to prepare training images in order to retrain the network: people addition (adding new people over time) and cameras addition (adding new samples from new cameras over time). *Training approaches* are 3 training methods used to retrain the network: fine-tuning, joint-training, and LwF. *Training stages* means training the network is performed in multi stages to mimic continuous learning.

4.1. Datasets

Market1501 [7] contains 1,501 identities which are captured from 6 different cameras in outdoor. 751 identities as used as training sets (including validation), and 750 identities are used as testing sets (32,668 gallery images, 500K images as distractor, and 3,368 query images). Each identity has approximately 10 samples. Each identity is present in at least two different cameras. Bounding boxes are detected automatically using DPM detector and then validated manually.

DukeMTMC4ReID [8] is a subset of DukeMTMC dataset and contains 1,404 identities which are captured from 8 different cameras in outdoor. 702 identities as training sets (including validation), and 702 identities as testing sets (177,661 gallery images and 2,228 query images). Each identity appears in more than two cameras, but 409 identities only appear in one camera acting as distractor. Automatic person detection is performed using DPM and Open Pose and manual annotation. Compared to Market1501, this dataset has more samples per identity which is at least 15 samples.

The training dataset is distributed into subsets of equal sizes to perform incremental training in stages. For each training stage, current training subset is treated as new tasks and previous training subsets are treated as old tasks. For testing, gallery and query images are taken from training sets, 20% and 10% respectively instead of using testing sets originally provided by the dataset authors. This strategy is to evaluate if the model can perform well on both new and old tasks. All images are mutually exclusive, as each image is only used once for either training or testing. To increase the number of samples, data augmentation is performed based on methods presented in [1], including random flip horizontal, random crop, and random erasing. These methods can improve the model's performance by 2% compared to common methods.

4.2. Training and testing implementation

The proposed model is implemented using *Tensorflow* with GPU support on NVIDIA GeForce RTX 2080 Ti with 8 GB memory and CUDA 10.0.

Training: For each training, the model is trained using the same hyper parameters. Batch size is 32 but smaller when using LwF. Learning rate is 0.05, which is experimentally set, and scheduled to decay by a factor of 0.1 every 40 epochs by a decay weight of 0.0005. Number of epochs is 60 and the optimizer uses stochastic gradient descent with momentum of 0.9 using *Nesterov momentum*.

Testing: Unlike how existing works evaluate the models, this experiment's testing is based on the same person instead of different people. The idea is to evaluate if the model can recognize people, which have been learnt either as the new or old tasks as described in Section 4.1. However, for benchmarking with existing works, the model is evaluated based on different people.

Evaluation metrics: average precision (*AP*), mean average precision (*mAP*) and training time are the metrics used to evaluate the models' performance. *AP* is computed from its precision-recall curve. *AP Rank-i* denotes the average precision based on the result of the *i-th* rank (from 1 to 10). Only rank 1, rank 5 and rank 10 are presented in the results section due to limited space. *mAP* is calculated as the mean of *AP* across all queries in all ranks.

5. Experiment results

Performance degradation on old tasks over time when old dataset is unavailable: Summarized from Table 1, Figure 3 shows how fine-tuning performs every time the network is retrained without old images. Catastrophic forgetting problem starts from Stage 2 as shown by reduced Rank-1 performance on old tasks by 15% on both datasets. For example, the performance is 0.47 in Stage 1 but drops to 0.37 after the network is retrained in Stage 2 on Market1501. Similarly, the performance gets worse from 0.35 to 0.22 in Stage 2 on DukeMTMCReID.

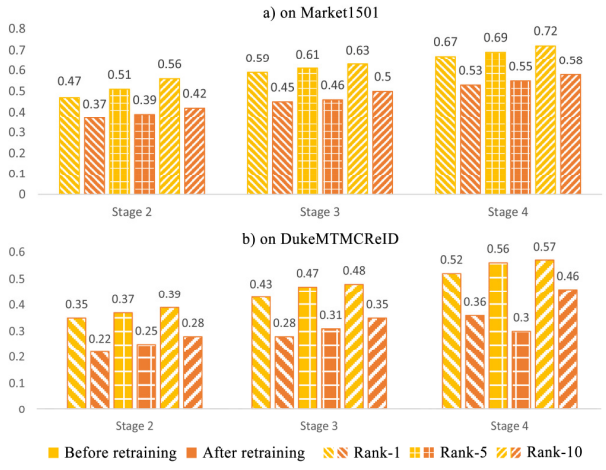

Figure 3: Performance degradation in fine-tuning (when using people addition strategy).

Performance comparison between training methods and the trade-offs: Overall, joint-training constantly maintains performance on both old and new tasks every time the network is retrained as seen in Table 1 and 2. The Rank-1 performance difference between old and new tasks is more or less up to 0.02 on both datasets. LwF and fine-tuning can also maintain the Rank-1 performance on new tasks over time slightly lower than joint-training by 0.05 and 0.07 respectively on both datasets. All training methods show increasing Rank-1 performance trend on new tasks between 0.02-0.18 over time. We argue that increasing the number of samples used by progressive training can result in a cumulative knowledge, which enables the network to learn features better.

While joint-training can maintain performance on old tasks, fine-tuning keeps forgetting the old tasks approximately by 29% on Rank-1 performance every time the network is retrained. By using new tasks' data, LwF can reduce the forgetting problem by 11% on both datasets compared to fine-tuning. Although LwF cannot outperform joint-training's performance, at least it can reduce the Rank-1 performance margin by 19% approximately on both datasets. For example, joint-training can achieve 0.65 on old tasks' Rank-1 performance in Stage 2 on Market1501. Fine-tuning can only achieve 0.37 but LwF can achieve 0.46. Figure 5 shows some example how LwF performs on old tasks. It starts losing its gained knowledge during continuous training.

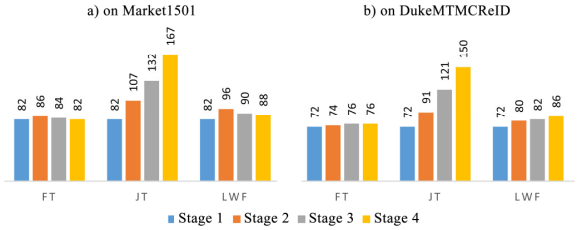

Figure 4: Training time comparison (in minutes) when using people addition strategy. Numbers are rounded up.

Although joint-training can lead the performance on both tasks, it requires much longer training time compared to other methods. Joint-training requires approximately 1.3 times longer than its previous training stage whereas fine-tuning and LwF require relatively constant training time regardless of training stage, as seen in Figure 4. The increasing number of training images in joint-training

(a) on Market1501

	Rank-1						Rank-5						Rank-10						mAP					
	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT
S1	NA	0.47	NA	0.47	NA	0.47	NA	0.51	NA	0.51	NA	0.51	NA	0.56	NA	0.56	NA	0.56	NA	0.55	NA	0.55	NA	0.55
S2	0.37	0.59	0.65	0.65	0.46	0.60	0.39	0.61	0.66	0.67	0.50	0.63	0.42	0.63	0.70	0.69	0.54	0.66	0.39	0.61	0.68	0.68	0.50	0.66
S3	0.45	0.67	0.74	0.74	0.54	0.69	0.46	0.69	0.77	0.76	0.58	0.71	0.50	0.72	0.77	0.78	0.61	0.73	0.47	0.69	0.76	0.77	0.58	0.74
S4	0.53	0.76	0.81	0.82	0.62	0.77	0.55	0.78	0.84	0.84	0.65	0.79	0.58	0.80	0.84	0.84	0.68	0.82	0.55	0.78	0.83	0.84	0.65	0.82

(b) on DukeMTMCReID

	Rank-1						Rank-5						Rank-10						mAP					
	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT
S1	NA	0.35	NA	0.35	NA	0.35	NA	0.37	NA	0.37	NA	0.37	NA	0.39	NA	0.39	NA	0.39	NA	0.39	NA	0.39	NA	0.39
S2	0.22	0.43	0.49	0.49	0.33	0.46	0.25	0.47	0.52	0.52	0.37	0.49	0.28	0.48	0.53	0.54	0.39	0.52	0.25	0.46	0.51	0.53	0.36	0.52
S3	0.28	0.52	0.58	0.59	0.39	0.54	0.31	0.56	0.60	0.61	0.42	0.58	0.35	0.57	0.62	0.62	0.45	0.60	0.31	0.55	0.60	0.62	0.42	0.60
S4	0.36	0.61	0.68	0.68	0.46	0.63	0.30	0.64	0.69	0.70	0.49	0.66	0.46	0.67	0.71	0.72	0.52	0.71	0.37	0.64	0.69	0.71	0.49	0.70

Table 1: Performance comparison on old tasks (OT) and new tasks (NT) using people addition strategy. FT JT LwF. Si denotes training at stage i. NA stands for "Not Available".

(a) on Market1501

	Rank-1						Rank-5						Rank-10						mAP					
	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT
S1	NA	0.69	NA	0.69	NA	0.69	NA	0.71	NA	0.71	NA	0.71	NA	0.73	NA	0.73	NA	0.73	NA	0.73	NA	0.73	NA	0.73
S2	0.50	0.72	0.77	0.78	0.60	0.75	0.52	0.74	0.81	0.80	0.63	0.77	0.55	0.76	0.81	0.81	0.65	0.79	0.52	0.74	0.80	0.81	0.63	0.80
S3	0.52	0.73	0.80	0.80	0.61	0.77	0.54	0.76	0.82	0.81	0.64	0.79	0.57	0.78	0.82	0.82	0.66	0.81	0.54	0.76	0.81	0.82	0.64	0.82
S4	0.55	0.77	0.82	0.83	0.64	0.79	0.56	0.79	0.84	0.84	0.67	0.81	0.66	0.81	0.84	0.85	0.69	0.83	0.59	0.79	0.83	0.85	0.67	0.84

(b) on DukeMTMCReID

	Rank-1						Rank-5						Rank-10						mAP					
	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT	OT	NT
S1	NA	0.55	NA	0.55	NA	0.55	NA	0.58	NA	0.58	NA	0.58	NA	0.61	NA	0.61	NA	0.61	NA	0.60	NA	0.60	NA	0.60
S2	0.35	0.58	0.64	0.64	0.46	0.61	0.38	0.60	0.65	0.65	0.48	0.61	0.41	0.63	0.69	0.68	0.52	0.66	0.38	0.60	0.66	0.67	0.49	0.66
S3	0.37	0.59	0.66	0.66	0.47	0.63	0.42	0.64	0.66	0.67	0.50	0.65	0.44	0.67	0.71	0.72	0.53	0.69	0.41	0.63	0.68	0.69	0.50	0.69
S4	0.40	0.62	0.68	0.69	0.49	0.65	0.44	0.65	0.70	0.70	0.52	0.66	0.46	0.68	0.73	0.73	0.55	0.71	0.43	0.65	0.70	0.72	0.52	0.70

Table 2: Performance comparison on old tasks (OT) and new tasks (NT) using cameras addition strategy. FT JT LwF. Si denotes training at stage i. NA stands for "Not Available".

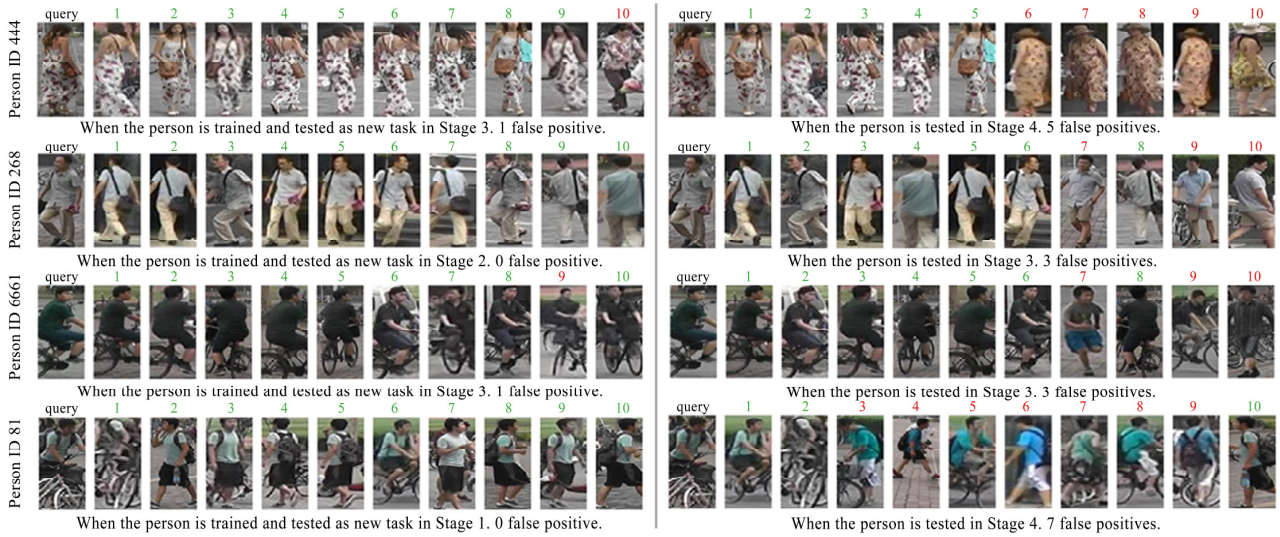

Figure 5: Some examples of testing result on Market1501 in different stages (when using LwF). ■ correct ■ incorrect person.

creates an inclining trend on training time, which clearly affects the training time due to more time required for extracting training images' features and updating weights more frequently during training. For example, joint-training requires 167 minutes to train the network in Stage 4 on Market1501, which is nearly two times slower than fine-tuning and LwF, 82 and 88 minutes respectively In addition, LwF requires a bit more time to train the network compared to fine-tuning, thanks to its technique for predicting outputs on old network and use of smaller batch size used, which will update weights more frequently.

People addition vs cameras addition comparison. Cameras addition strategy achieves the best performance in first stage training compared to people addition, which is up 22% in difference on both datasets as seen in Table 1 and Table 2. For example, cameras addition strategy can achieve 0.69 and 0.55 on Rank-1 performance on Market1501 and DukeMTMCReID respectively, whereas people addition strategy can only achieve 0.47 and 0.35. However, cameras addition strategy does not have that much performance improvement over time as the performance can only improves by 3% approximately whereas people addition strategy can give improvement by 9% approximately on both datasets.

6. Discussions

LwF can alleviate forgetting problem. Although fine-tuning can perform well on new tasks, the result confirms that it suffers from performance degradation on old tasks over time, every time the network is retrained due the absence of old datasets. It confirms that the use of old datasets has a strong role in retaining the gained knowledge, which is why joint-training requires both datasets during training. LwF can preserve its performance on old tasks comparably without using old datasets by 11% approximately compared to fine-tuning. Knowledge distillation plays an important role to predict the outputs of old tasks similar to the old network without the presence of old datasets.

Best training strategy for long-term performance. To train a network for long-term performance, we suggest combining the training strategies: training with people addition at the beginning to give initial knowledge then followed by cameras addition strategy to refine the model.

Training the network with large number of people at beginning plays important role because it can give enough features to the network to learn as initial knowledge. In addition, although our results show that cameras addition strategy gives significant initial performance instead of people addition strategy, we argue that it is due to large number of training images used in Stage 1. We found that Stage 1 has more samples than other stages, which makes the model learns the feature enough at the beginning. When the model has learnt the features enough from new people, the performance can be improved by providing more discriminative features from others cameras when new cameras introduced. We believe that this strategy can make the model more robust to any variations in long term as new cameras may bring more discriminative features representing new variations among multi cameras.

Our result demonstrates that continuous learning can improve overall performance over time. The state-of-the-art results on the same datasets have been shown up higher by 0.19 and 0.28 on Market1501 and DukeMTMCReID respectively, but their methods are focusing on training-at-once approach which requires the data available at the same time and cost more storage and processing time. Continuous learning can support incremental learning as we keep on adding more data in future, supporting incremental improvement and scalability.

More discriminative samples is more important than more samples in dataset: Market1501 gives much better

overall performance by 14% in approximation compared to DukeMTMCReID in any training approaches. Arguably, Market1501 offers more various visual appearance features between each identity, which can help the network to learn discriminative features better. Although, DukeMTMCReID offers more number of samples for each identity but it cannot beat the performance on Market1501 as more discriminative samples plays more important role towards performance.

7. Conclusions and Future Work

This paper extends the benefit of LwF to perform person re-identification task in continuous learning. Comprehensive experiments on different datasets with various approaches are presented to evaluate how effective LwF can maintain performance on both tasks over time compared to existing methods. The findings confirm that LwF provides a viable solution for continuous learning in person re-identification by outperforming both fine-tuning for better knowledge preservation by 11% and joint-training for faster training time.

Acknowledgement

The authors would like to thank Southern Cross University and Queensland Airports Limited (QAL) for the support given during this work.

References

[1] Z. Zhong, L. Zheng, Z. Zheng, S. Li, and Y. Yang, "Camstyle: a novel data augmentation method for person re-identification," *IEEE Transactions on Image Processing,* vol. 28, no. 3, pp. 1176-1190, 2019.

[2] J. Donahue *et al.*, "Decaf: A deep convolutional activation feature for generic visual recognition," in *International conference on machine learning*, 2014, pp. 647-655.

[3] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580-587.

[4] R. Caruana, "Multitask learning," *Machine learning,* vol. 28, no. 1, pp. 41-75, 1997.

[5] M. McCloskey and N. J. Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem," in *Psychology of learning and motivation*, vol. 24: Elsevier, 1989, pp. 109-165.

[6] Z. Li and D. Hoiem, "Learning without forgetting," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 40, no. 12, pp. 2935-2947, 2018.

[7] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, and Q. Tian, "Scalable person re-identification: A benchmark," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1116-1124.

[8] M. Gou, S. Karanam, W. Liu, O. Camps, and R. J. Radke, "DukeMTMC4ReID: A large-scale multi-camera person re-identification dataset," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 10-19.

[9] D. Wu *et al.*, "Deep learning-based methods for person re-identification: A comprehensive review," *Neurocomputing,* 2019.

[10] P. Dollár, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: A benchmark," presented at the IEEE Conf. Computer Vision and Pettern Recognition, 2009.

[11] W. Van Ranst, F. De Smedt, J. Berte, and T. Goedemé, "Fast simultaneous people detection and re-identification in a single shot network," in *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2018, pp. 1-6: IEEE.

[12] J. Wang, X. Zhu, S. Gong, and W. Li, "Transferable joint attribute-identity deep learning for unsupervised person re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2275-2284.

[13] C. Zhao, K. Chen, Z. Wei, Y. Chen, D. Miao, and W. Wang, "Multilevel triplet deep learning model for person re-identification," *Pattern Recognition Letters,* vol. 117, pp. 161-168, 2019.

[14] H. Luo, W. Jiang, X. Zhang, X. Fan, J. Qian, and C. Zhang, "AlignedReID++: Dynamically Matching Local Information for Person Re-Identification," *Pattern Recognition,* 2019.

[15] R. M. French, "Dynamically constraining connectionist networks to produce distributed, orthogonal representations to reduce catastrophic interference," *network,* vol. 1111, p. 00001, 1994.

[16] Y. LeCun *et al.*, "Backpropagation applied to handwritten zip code recognition," *Neural computation,* vol. 1, no. 4, pp. 541-551, 1989.

[17] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," presented at the Proc. 28th Int. Conf. Neural Inf. Process. Syst. Workshop, Montreal, Canada, 2014.

[18] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "Fitnets: Hints for thin deep nets," presented at the Proceeding of the International Conference on Learning Represntations (ICLR), 2015.

[19] T. Chen, I. Goodfellow, and J. Shlens, "Net2net: Accelerating learning via knowledge transfer," presented at the Proceedings of the International Conference on Learning Representations (ICLR), 2016.

[20] S. Hou, X. Pan, C. Change Loy, Z. Wang, and D. Lin, "Lifelong learning via progressive distillation and retrospection," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 437-452.

[21] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "icarl: Incremental classifier and representation learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2001-2010.

[22] K. Shmelkov, C. Schmid, and K. Alahari, "Incremental learning of object detectors without catastrophic forgetting," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3400-3409.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770-778.