Investigating Fast Re-identification for Multi-camera Indoor Person Tracking

Andrew Tzer-Yeu Chen[*], Morteza Biglari-Abhari, Kevin I-Kai Wang

Embedded Systems Research Group, Department of Electrical and Computer Engineering, The University of Auckland, Auckland, New Zealand

Abstract

In person tracking applications involving multiple cameras, person re-identification is an important step for ensuring accurate tracking of individuals as they move between camera views. However, changes in camera parameters and environmental conditions can make re-identification challenging. This is especially difficult in resource-constrained environments, as is often the case in many real-world intelligent applications. In this paper, we explore dimensionality reduction, metric learning, and classification for achieving re-identification in a computationally efficient way. We report that the covariance metric transformation is a sufficient distance metric for achieving good linear separability between identity classes, and produces better results than more complex approaches across two re-identification datasets. We also explore one-shot learning methods for performing classification, show that our Sequential k-Means algorithm outperforms other fast one-shot learning approaches, and discuss parameter tuning to improve accuracy.

Keywords: Computer Vision, Person Tracking, Person Re-identification, Metric Learning, Classification, Camera Surveillance

1. Introduction

Continuous traceability of human locations in real-time is a fundamental requirement underpinning many intelligent applications involving behaviour analysis, such as smart home environments, pedestrian management, retail market research, and law enforcement. While camera-based approaches can offer highly accurate location estimation, they often suffer from a limited field of view, limited inherently by the aperture and orientation of the cameras as well as by obstacles that obscure the object(s) of interest. Such limitations greatly impede the practicality and effectiveness of using camera-based approaches for human behaviour analysis as a part of many intelligent applications. Mitigation of these issues generally requires the use of multiple cameras, either overlapping or non-overlapping, at different camera positions and angles to capture a wider area of the indoor space. In order to track people, we need to be able to identify that one person in one camera view is the same as a person previously seen in another camera view, so that we can connect those two instances to the same identity and track their movements through the physical space. There are two ways of approaching this problem: either a spatio-temporal model that uses the topology of the cameras to connect tracks based on when and where objects have entered and exited camera views, or a re-identification approach that matches the visual appearances of individuals across multiple camera views [1]. In our wider research project, we are working on combining these two methods together to achieve higher accuracy rates, but in this paper we focus on improving the accuracy of appearance-based re-identification.

Our research focuses on indoor occupancy studies, where we want to be able to answer questions about how people are using physical spaces. This type of application is sometimes called video analytics rather than surveillance to distinguish the purpose of the camera network from law enforcement or public safety applications. For example, a co-working space may be interested in knowing how their office space is being used, such as which spaces are more popular, or how the usage patterns change over time. Identifying the different types of users and how they move

[*] Corresponding Author

Email addresses: `andrew.chen@auckland.ac.nz` (Andrew Tzer-Yeu Chen), `m.abhari@auckland.ac.nz` (Morteza Biglari-Abhari), `kevin.wang@auckland.ac.nz` (Kevin I-Kai Wang)

Algorithm 1 Pseudocode to describe the algorithm flow shown in Figure 1

```
Receive an input image $I$ from the camera
Run DPM on $I$ to extract all people $p$ into a set $P$
for each person $p$ in set $P$ do
    Run feature extraction to get the Colour ($H_{col}$) and LBP ($H_{LBP}$) Histograms
    Concatenate $H_{col}$ and $H_{LBP}$ to form the feature vector $fv_p$
    Pass $fv_p$ through a dimensionality reduction matrix learned from PCA to produce $rv_p$
    Pass $rv_p$ through a learned transformation matrix from a metric learning algorithm to produce $lv_p$

    for each class $c$ in the identity gallery $C$ do
        Compare $lv_p$ and $lv_c$ by calculating the Euclidean distance $d_{p,c}$
    end for
    Form a set $D$ of the distances $d_{p,c}$ for each class $c$
    Sort $D$ to find the lowest distance $d_{p,c}$ and declare class $c$ to be the matching identity
    Update the class model $lv_c$ using the one-shot learning algorithm
    Output $c$ as the class/identity label output
end for
```

throughout the space may also help inform decisions about how to allocate and design the space, and communicate with smart devices such as air conditioning systems or coffee machines. Areas with minimal staffing or large areas are difficult to monitor with human staff or with a single camera. Therefore, we cannot simply run person detection on some camera footage - we need to be able to at least re-identify, and potentially track, the individuals to ensure that we do not double count individuals. Similar technology can be used in brick-and-mortar retail stores to get insights into customer behaviour to enable A/B testing and inform decisions around product placement, loyalty programs, and staff scheduling.

It is desirable to develop a re-identification approach that is computationally efficient and can be executed on embedded devices with limited resources, suitable for indoor spaces. However, this problem is made challenging by the fact that external environmental factors, such as lighting and noise generated by intermittently moving background objects, can differ between different camera locations, causing the visual appearance of the individual to vary between camera views. Variation in camera angles and positions can also lead to the same object having different appearances between cameras - for example, one camera may be positioned so that it sees the person walking towards the camera, while another camera in the network sees the person walking away from it, causing the perceived appearances to vary significantly. The timing relationships and positions of the cameras (i.e. the topology) can be difficult to learn [2].

We can divide the person re-identification task into several components. Once a person has been detected in a camera view, we need to represent that person in some way. One option is to pass the pixel information of the person directly into a deep convolutional neural network (CNN) [3] for classification. In the case of [4], person detection/search and re-identification are done jointly on raw images with a single CNN. However, CNN approaches require a significant amount of training data for each identity and are computationally very expensive, and still face many challenges in real-time embedded implementations. A generally more preferable and simple solution is to extract some higher-level features, manipulate that data to make it more separable, and then classify those features into person identity classes [5, 6]. This also better supports one-shot or unsupervised learning methods in scenarios with limited training data, where the system is expected to learn as it processes real data. Importantly, any method needs to be scalable, since in a real-world scenario we might reasonably expect to have to manage hundreds of possible identities.

In this paper, we present a qualitative and quantitative analysis of different methods used for two key steps in our person re-identification process: metric learning for transforming the feature vectors so that they are easier to classify (as part of the feature vector extraction), and classification of those transformed feature vectors in order to identify the individual. The steps conducted in the overall process are shown in pseudocode in Algorithm 1 and as a flowchart in Figure 1. Our key contributions are that a) we show that using the covariance matrix calculation as a transformation function actually does a sufficiently good job of creating linearly separable feature vectors, and outperforms many more sophisticated metric learning methods, and b) we show that the unsupervised Sequential k-

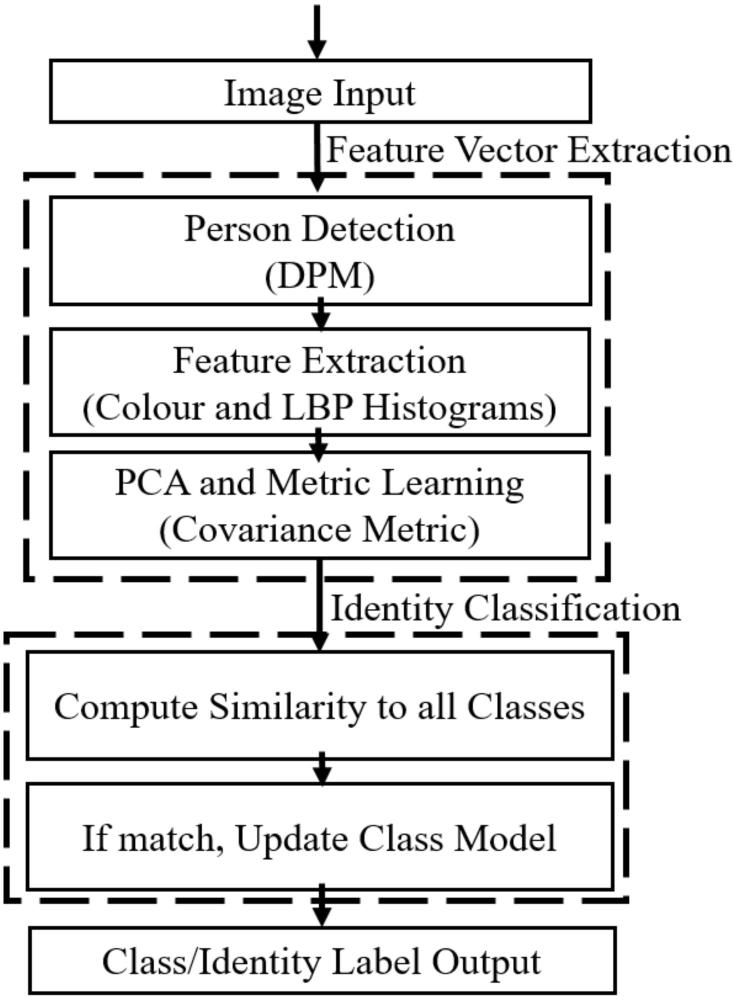
Image Input
Feature Vector Extraction
Person Detection (DPM)
Feature Extraction (Colour and LBP Histograms)
PCA and Metric Learning (Covariance Metric)
Identity Classification
Compute Similarity to all Classes
If match, Update Class Model
Class/Identity Label Output

Figure 1: The feature extraction and identity classification processes

Means method described in our previous conference papers [7, 8] achieves very good results for classifying identities from feature vectors under our problem setting where there is very little training data available, represented in the worst case as a one-shot classification challenge.

In Section 2, we present the main dataset that we used, as well as the first processing steps of feature extraction and dimensionality reduction. In Section 3, we discuss metric learning and comprehensively compare a number of different methods on two datasets. In Section 4, we discuss a number of different supervised, semi-supervised, and one-shot learning classification methods, presenting results in terms of accuracy and computation time. We conclude the paper and discuss future work in Section 5. Related works are referred to throughout the paper in each section.

2. Experimental Setup

2.1. Test Environment

The standard formulation of a re-identification challenge, such as in ViPER and DukeMTMC4ReID [9], involves having a large number of identities and a small number of samples for each identity in a gallery, and then trying to match a new source image of a person to an existing target image in the gallery. In contrast, with real-world indoor scenarios such as retail stores or office spaces, we are likely to encounter a smaller number of identities across a given

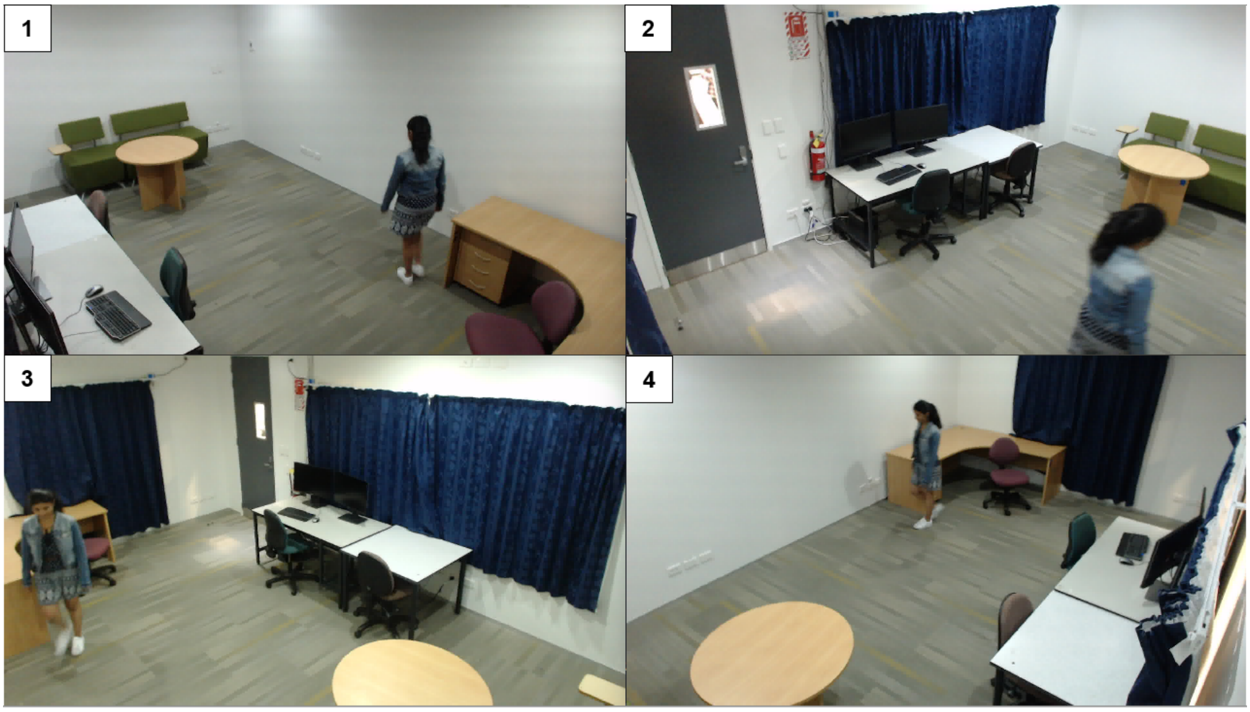

Figure 2: Side-by-side views from the four cameras in the dataset

time period, and see them within view for a much longer time period, which allows us to collect more samples of each identity [5, 10]. We also assume that with the assistance of an auxiliary sensor, such as an infra-red detector or a Radio Frequency Identification (RFID) authenticator at a turnstile, we are able to detect when a person enters the area covered by the camera network for the first time, so that we can capture a frame and use that as an anchor for a new identity class in a one-shot setting.

Evaluation of different approaches requires a standardised dataset for fair comparison. A standard re-identification formulation is perhaps not as suitable for our requirements - rather than worrying about whether we can successfully re-identify one individual with only one or a few valid gallery samples, we are more interested in our ability to continually re-identify a person over time with an increasing number of gallery images, both within and across different camera views. Instead of using a dataset like ViPER, which only has a single source and single target image for each identity, we prefer person tracking datasets that can be used to extract people from each frame and build up a gallery over time, providing the algorithm with more information as we progress. [11] includes a discussion of existing person tracking datasets, but most of these datasets only contain bounding boxes and do not include location data or ground truth topology information between the cameras. Recent datasets such as DukeMTMC [11] and 3DPeS are perhaps the closest to our use case, but both of these datasets are recorded in an outdoor context where individuals appear smaller than in an indoor context, and the lighting conditions are generally more consistent across the same camera view. As a result, we primarily tested our approaches on an internally created dataset called UoA-Indoor, using four overlapping camera views as shown in Figure 2.

We set up the room with standard office furniture as shown in Figure 3 to emulate an indoor office environment. Footage was recorded at a resolution of 1920x1080 at 15 frames per second, using a script to initialise and synchronise all four cameras so that they record at the same time. For the purposes of fair analysis in this paper, we downsampled the raw footage to 640x480 to reduce computation time. We used consumer-grade Logitech C920 webcams for the collection of footage. The use of common webcams over more advanced camera equipment was an intentional decision to allow the setup to be more accessible, low-cost, and reproducible, as well as for any developed algorithms to be more robust and more likely to work with low-cost or old cameras in other scenarios. While it has been argued

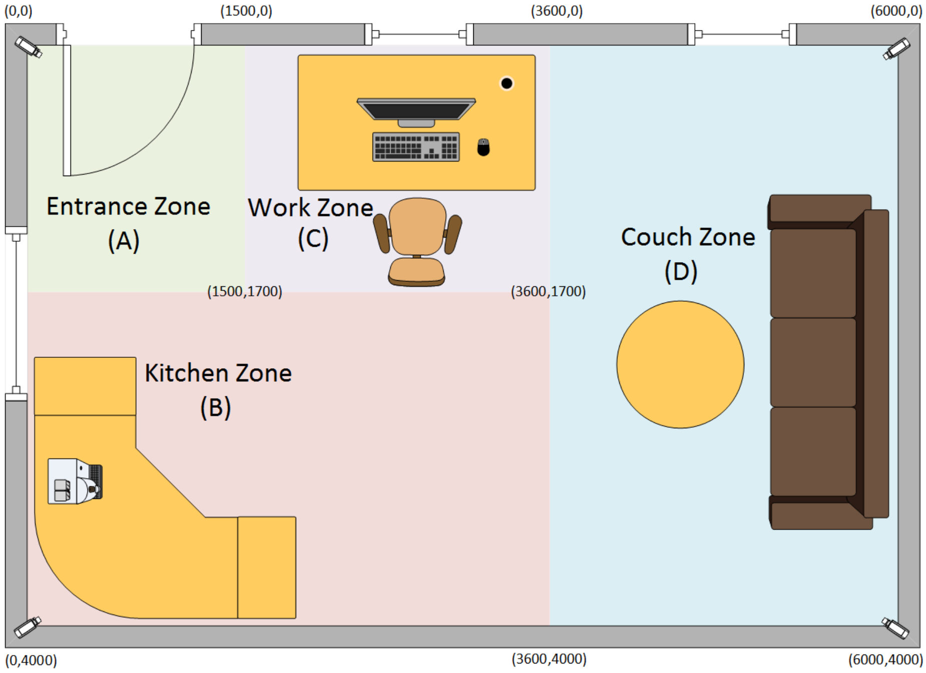

Figure 3: A map of the UoA-Indoor dataset test environment with a camera in each corner (co-ordinates given in mm)

that using stereo cameras or other cameras that provide some measurement of depth (e.g. RGBD, infra-red, etc.) can lead to higher accuracy levels, single view cameras are still currently the most common in existing surveillance infrastructure, allowing single view camera-based algorithms to be more easily integrated. We currently have over 2 hours and 45 minutes of footage collected with 19 different identities appearing across over 150,000 frames. We expect to add to this dataset over time as we create more complex and realistic scenarios with multiple people. We also hope to capture footage for other indoor spaces such as corridors and foyers, and in real-world retail spaces to make the dataset closer to our end target application. More details about the dataset and the publicly available annotation tool that we created are available in [7].

After the creation of the dataset, we used the OpenCV implementation of the Deformable Parts Model (DPM) detector to extract all instances of people in the dataset, and manually labelled the identities. Examples of samples from two identity classes are shown in Figure 4, demonstrating some of the challenges that are posed by the variability in this dataset. In particular, readers should note the significant differences in camera and subject pose, partial obscuration by objects naturally appearing in the environment, motion blur from low-quality cameras, and the presence of multiple lighting sources causing significant colour changes and saturation. Importantly, these issues appear not only between cameras but also within camera views, making learning a single transfer function between the different camera views very challenging. Our re-identification dataset contains approximately 28,000 samples across 19 identities with between 750 and 2640 samples per identity, as well as approximately 13,000 hard negative samples, across all four camera views.

2.2. *Feature Selection*

The main reason for using DPM instead of a more recent or faster person detection algorithm is that it allows us to extract parts for each person, corresponding to parts of the body such as the head, arms, hips, and feet. While others have used simpler methods such as horizontal stripes [5] or approximate blobs [12] to segment people for the purposes of feature extraction, using better segmented body parts provides a better proxy for how humans identify individuals at a distance [13]. For example, a human might identify a person by saying that they have brown hair and are wearing a red jacket, which could be approximated by a computer identifying that the head part of the human has a significant

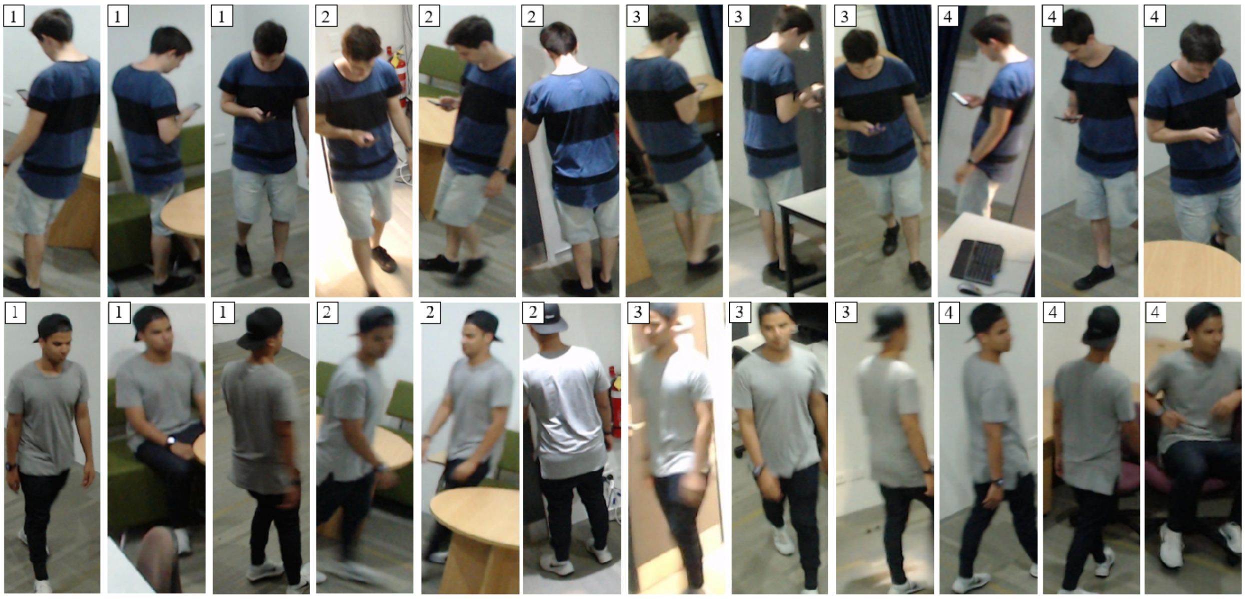

Figure 4: Samples from the UoA-Indoor re-identification dataset for two classes extracted with DPM, with the camera view number shown in the top left of each sample

portion of brown and the chest part of the human is predominantly red. This type of features has also been called soft biometrics more recently [14].

Similar to [15], we extract colour and texture descriptors for each part, using a 15x16-bin HSV 2D-colour histogram (across hue and saturation), a 3x16-bin RGB colour histogram, and a 16-bin local binary pattern (LBP) histogram for texture. These features can be computed very quickly for each part, and also provide good discriminative strength [16]. Additionally, using DPM, we can approximate whether a part of the body is obscured or out of frame by checking the matching/similarity score. This allows us to ignore the features extracted for invalid parts as that would contribute towards erroneous matching. We can then concatenate the colour and texture features together for each of the eight parts to form a 2432-dimension feature vector for the detected individual. While it is possible to use more histogram bins and have a longer feature vector, the increase in the number of dimensions may not lead to a sufficient accuracy increase to justify the added computation cost of extracting and processing more data for each detected person, so a good balance should be found.

We then use Principal Components Analysis (PCA) as an unsupervised method of determining the most important dimensions of the feature vectors in terms of variation - dimensions that have low variation are likely to correspond to background parts of the image that are common between all classes, while high variation dimensions are likely to help with separability between classes. This approach is also used by [5] for a real-world person re-identification application. Reducing the number of dimensions to a relatively small number of most important dimensions lowers the computation time for metric learning and classification as there is less data to process. Additionally, for some algorithms, reducing the number of dimensions can improve accuracy as uniformly weighted noisy and misleading dimensions can be removed, leaving the more important discriminative dimensions. For example, dimensions that correspond to the histogram bins that represent textures mostly only found in the background, such as on a wooden table, would be removed as they are likely to have the same values for two different people standing in front of a wooden table, and therefore would not help with inter-class classification. Once the most important dimensions have been established by showing enough (unlabelled) training samples offline, this can be applied to all future feature vectors without any additional training. As the number of resultant dimensions increases, the increase in classification computation time is generally exponential, while the accuracy improves diminishingly or even decreases as more dimensions are introduced, as shown in Figure 5. This figure shows that between 32 and 64 dimensions were generally sufficient for good separability during subsequent classification.

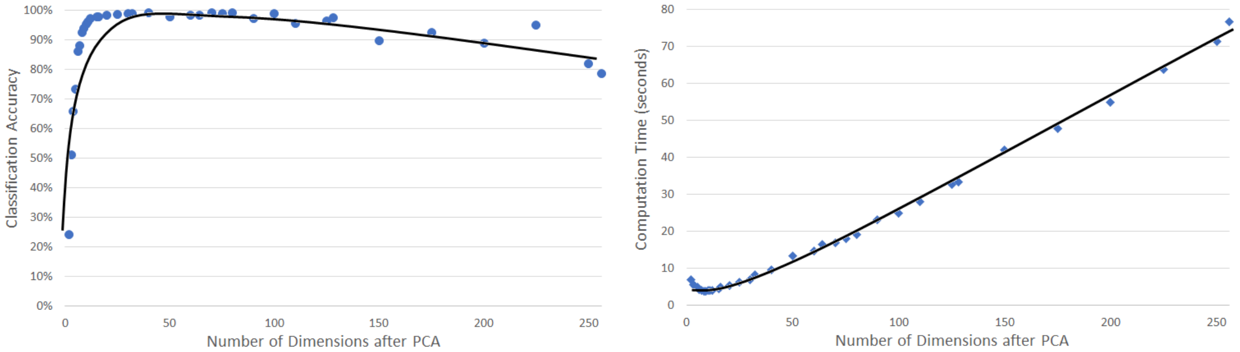

Figure 5: Classification accuracy (using 5-fold cross-validation) and computation time (training and testing time combined, per fold) from a linear one-vs-all SVM classifying approx. 28,000 feature vectors across 19 classes with between 2 and 256 dimensions after PCA and a baseline covariance metric learning transformation

3. Metric Learning

After PCA, we have a series of feature vectors based on the features/variables/dimensions with the highest level of variation. However, these vectors are often not linearly separable, which makes classification more challenging, and no effort is made to keep samples from the same class together. While it is common to use non-linear kernels such as Radial Basis Functions (RBF) to achieve non-linear classification, this can be very computationally expensive, and can still find some data distributions very challenging. Metric learning (also known as distance metric learning) allows us to learn a distance metric, generally a Mahalanobis distance metric, that optimises towards establishing high inter-class (between-class) variation and low intra-class (within-class) variation. Metric learning has been shown to be a useful pre-processing step for many machine learning approaches, and has been shown to improve the accuracy of person re-identification [5, 17] by transforming the vectors so that they are more linearly separable into identity classes, which has the effect of increasing the accuracy and reducing the subsequent training and testing times required by the classification algorithms. For more detailed information about metric learning methods and their applications, particularly in the context of processing feature vectors, we refer readers to the survey technical report [18].

By including annotated/labelled images of detected people across a number of different camera views, the aim is to use the metric learning algorithm to learn how to minimise the impact of the different camera characteristics by minimising intra-class variation, where the class is a single identity that has samples across multiple camera views. This is in contrast to approaches that explicitly learn a transfer function that models the change in parameters between two camera views [19] so that the appearance of a person in one camera view can be modified to be more similar to what we might expect if they had appeared in another camera view. The challenge for real-world re-identification is that we have to learn a metric that is applicable to classes that we may not have seen at training time, as new people enter and exit the monitored environment. Thus far, the literature has not shown there to be a "universal" distance metric that can be applicable in all cases, as it is common for the learned metric to be dependent on camera parameters, external environmental factors, and variations in background [20]. While the optimum metric could be learned if many samples of all classes were available at training time, in reality we need to find a metric based on a small number of training classes that can then be used on other new classes at test time, similar to how the important dimensions are estabilshed through PCA. This is further complicated by the fact that there are many, many different metric learning algorithms, and there is no consensus yet on which algorithm is the most suitable for person re-identification.

3.1. Algorithms

In this work, we test a selection of the most popular metric learning algorithms, which have been validated in many applications over time. All of these algorithms have also been shown to produce good results in person re-identification or similar image/signal processing applications, hence justifying their inclusion in our experiments. We provide a brief description of each of the tested algorithms here, but readers should refer to [21] for the references on each algorithm. **Large Margin Nearest Neighbour (LMNN)** uses a k-Nearest Neighbours (kNN) approach to try and keep the closest points together in the same class, while separating classes with margins that are as large as possible using semidefinite programming. This approach is similar to an SVM in that there are similar goals to maximise linear margins between classes, except it is based on kNN instead of a linear kernel. In our experiments, we set k to be equal to the size of the smallest class. LMNN and its derivatives are some of the most popular metric learning algorithms, and have previously been applied to person re-identification. **Information Theoretic Metric Learning (ITML)** views the samples of each class as a Gaussian and attempts to minimise the differential relative entropy, essentially bringing the samples for the same class closer together and inherently creating larger distances between the classes. This approach does not require semidefinite programming or expensive eigenvalue computation, but may require additional linear constraints to work effectively. ITML and improved versions of ITML have been used for facial recognition/verification and person re-identification. **Sparse Determinant Metric Learning (SDML)** attempts to take advantage of the typically sparse nature of data points in high-dimensional feature spaces, using regularisation approaches to learn a compact distance metric in a computationally efficient way. However, since we perform PCA as a form of dimensionality reduction first, some of the sparsity assumptions about the ratio of samples to dimensions may not hold true. **Least Squares Metric Learning (LSML)** is designed to minimise the sum of squared residuals in order to improve clustering accuracy by using a simple gradient-descent approach. It is also designed to work in sparse dimension spaces, and has been shown to work well in scenarios with minimal supervision. LSML is applied to handwritten character recognition in the original paper, but we believe that it is likely to be suitable for other image-based matching or re-identification applications since sparsity is a common characteristic of some dimensions in the feature vectors. LSML has been used in diverse image classification applications and facial recognition. **Local Fisher Discriminant Analysis (LFDA)** solves a generalised eigenvalue problem to perform supervised dimensionality reduction, and thereby produce separable classes. In particular, it is targeted at scenarios with multimodal distributions where the same class is actually distributed across multiple clusters (or in other words, the cluster may be made up of multiple separable sub-clusters), which we might expect when drawing samples from multiple camera views. It has previously been applied in a person re-identification context. **Relevant Components Analysis (RCA)** reduces global unwanted variability by assigning larger weights to relevant dimensions and lower weights to irrelevant dimensions. RCA does this based on "chunklets", defined as a subset of points belonging to the same class, in order to identify intra-class variability and minimise the weighting for those dimensions. RCA has been applied to facial recognition. Lastly, **Neighbourhood Components Analysis (NCA)** is based on a kNN approach, similar to LMNN, except it maximises an accuracy score based on the training data by iterating over stochastic solutions. However, NCA faces substantial computation and memory challenges with datasets that have high-dimensionality and a large number of samples, and can become trapped by local maxima. Both RCA and NCA have been applied to handwritten character recognition and face recognition. We also briefly investigated the recently popular KISSME, but decided not to include it in testing as [22] showed that it produced similar results to LFDA on person re-identification datasets, with LFDA outperforming KISSME on smaller datasets with fewer identity classes. In addition to the metric learning approaches described above, we also included a **covariance metric transformation** as a baseline method for comparison, based on the original Mahalanobis work. The covariance metric essentially applies a standard covariance operation over the feature vector:

$$\Sigma_{ij} = \mu[X_i X_j] - \mu_i \mu_j \tag{1}$$

where X is the feature vector, μ is the mean, and i, j refer to the positions of elements within the vector. For the rest of this paper, when we refer to "metric learning algorithms", we include the covariance metric in this collective group, even though strictly speaking there is no learning involved in this calculation. It is also important to include a control case in our experiments, where no metric learning is applied at all, to show that metric learning does produce better results than doing nothing.

3.2. Experimental Results

3.2.1. UoA-Indoor Dataset

Using our dataset, we set up two experiments to test the effectiveness of different metric learning algorithms. In both cases, we use the feature vectors after PCA (32-dimensions) has been applied. Firstly, in Experiment A (shown in Table 1), we used 5-fold cross-validation (CV), where the metric learning algorithm had access to 80% of the samples across all classes as a training set. It should be noted that these samples come from all four camera views, so there may be significant intra-class variation from the different camera views that needs to be minimised. We then applied the learnt distance metric to all of the samples, and made the transformed training set available to a linear one-vs-all Support Vector Machine (SVM), and tested this with the remaining 20% of the transformed samples as a test set, evaluating the accuracy and timing across five folds. The samples used in the training and test set were selected uniformly randomly, rather than being in sequential time order. In all experiments, the accuracy is essentially the "Rank 1" result, i.e. only the highest ranked result is used for classification with no further re-ranking.

After determining which metric learning algorithms were most likely to be suitable for our use case, in our second experiment (Experiment B, shown in Table 2) we used 5-fold cross-validation to provide four random classes (21% of all classes) to the metric learning algorithm, applied the learnt distance metric to all samples across all classes, and then classified the remaining transformed fifteen classes with a linear one-vs-all SVM. For each class, we provide 20% of the transformed samples to the SVM for training, and use the other 80% for testing. Using 5-fold cross-validation in this way allows us to confirm that the effect of metric learning is not closely tied to the specific classes that are chosen for training and testing. This experiment differs from Experiment A and the work done in [17] in that not all classified classes are made available to the metric learning algorithm, so we are testing the generalisability of the learnt distance metric on different classes in our dataset. The methods that were considered unsuccessful in Experiment A were not included again in Experiment B.

In our experiments, we ideally want to find a metric learning algorithm that leads to high classification accuracy and low computation times, both in terms of metric learning and for classification. The metric learning time is the time taken for the metric learning algorithm to return a transformation matrix - the time required to apply that transformation matrix to the data is constant for the same number of samples and negligibly small, regardless of the metric learning algorithm. The SVM training time is the time taken for the SVM to learn - the time required to classify each sample of the testing data is also constant for the same number of samples regardless of the metric learning algorithm, since the SVM just evaluates which side of the hyperplanes the sample lies on. It is important to note that these times are provided for processing the entire dataset, not per frame or per detected person. The metric learning and SVM training times should be interpreted comparatively between the different methods rather than taken as universally applicable computation times, as they will otherwise vary significantly depending on the computation platform and dataset. All accuracies and computation times are derived from a desktop computer running Ubuntu 16.04 with an i7-6700 CPU with four 3.40GHz cores and 64GB of RAM. This is a far more powerful computational platform than we would expect to be available in low-cost resource-constrained environments, but is used as a preliminary testbench. Even then, a number of the metric learning algorithms crashed when the PC ran out of memory, indicating that they may not be suitable for our resource-constrained use case. In particular, LMNN and NCA were not able to complete execution on the test computer, consuming all of the computational resources and memory. Additionally, RCA crashed when the number of post-PCA dimensions increased to 64, so the results in the table are for RCA on 32-dimensional feature vectors. The code is written in Python with NumPy to improve computation speed, and makes use of the *metric-learn*[1] library for some of the metric learning algorithm implementations in order to accelerate our testing. In each experiment, there are 19 identity classes and a hard negative class containing erroneous background samples that are not people.

As shown in the results for Experiment A in Table 1, the use of an appropriate metric learning algorithm clearly makes a large impact on the linear separability of the feature vectors, improving the accuracy significantly. In the case where no metric learning is used, the SVM seems to classify all of the test set vectors as being part of the hard negative class, resulting in a baseline accuracy of 9.44%. ITML and SDML show similar results, indicating that they have failed to learn a meaningful metric and have become biased towards the class with the largest number of samples. When LSML, LFDA, or RCA are used, we see the accuracy rise to over 90% using the same underlying classification method. Calculating the simple covariance metric of the reduced feature vector also leads to a high classification accuracy.

There is also large variation in the metric learning times between the different approaches. This is mostly dictated by how many iterations the approach needs, and whether the error can be reduced sufficiently or if the algorithm reaches the maximum number of iterations before exiting. However, it is also interesting to observe the differences in SVM training time, which can be interpreted as a sign of how successful the metric learning algorithm is at producing

[1] https://github.com/metric-learn/metric-learn

Table 1: Experiment A: Metric Learning with 5-fold CV across all classes, evaluated with SVM on UoA-Indoor

Method	Accuracy (%)	Metric Learning Time (s)	SVM Training Time (s)
None	9.44	0	58.53
ITML	9.44	7.84	56.27
SDML	9.44	0.019	56.23
LFDA	90.25	1.03	19.67
RCA	96.92	0.0084	60.52
Covariance	99.65	0.011	5.98
LSML	99.72	3.82	5.99
NCA	N/A	N/A	N/A
LMNN	N/A	N/A	N/A

Table 2: Experiment B: Metric Learning with a subset of classes, evaluated on 5-fold CV with SVM on unseen classes on UoA-Indoor

Method	Accuracy (%)	Metric Learning Time (s)	SVM Training Time (s)
None	18.66	0	17.78
LFDA	55.01	0.49	1133.59
RCA	58.56	2.95	82.62
LSML	96.07	3.86	17.58
Covariance	98.82	0.0061	8.38

linearly separable classes, as it represents how easy or difficult it is for the SVM to optimise hyperplanes between classes. High training times are a sign that learning a suitable hyperplane has been very difficult for the SVM, and in some cases leads to a very low accuracy when the SVM has failed to improve over time. This is with the exception of RCA, where the end accuracy is relatively high, but it requires a high number of iterations, as shown in the very high training time.

In terms of optimising towards high accuracy and low computation time, it is clear that LSML should be the winner in Experiment A, but the covariance metric performs very similarly with a much lower metric learning time (since there is no iterative learning process involved). These results are reinforced by the results of Experiment B, shown in Table 2. In this case, we see that the accuracies generally drop in comparison to Experiment A, which is to be expected since the metric learning algorithm has not seen all possible classes. The accuracy for the case with no metric learning at all increases since there are fewer classes (and samples) in the SVM train/test sets. The SVM training time increases, sometimes very significantly, since the learnt metric is likely to be less suitable for classes that have not been seen by the metric learning algorithm, and therefore the feature vectors are harder to separate. Experiment B shows that the high accuracies achieved by LFDA and RCA in Experiment A are highly dependent on the samples seen at training time, suggesting that these approaches tend to overfit. This may be suitable for some applications where all classes are available at training time, but this is usually not the case in the real-world. LSML is left as the only learning approach that achieves a high accuracy when the distance metric is applied to new data containing new classes, but surprisingly the covariance metric, without any learning, achieves an even higher accuracy.

LSML and the covariance metric transformation produce similar results because in LSML, the covariance metric (i.e. converting to the Mahalanobis Distance) is used as an initial guess, which is then further iterated upon to reduce the error - since the covariance metric is already a good guess, further iterations only lead to marginal gains. However, a 0.07% increase in accuracy shown in Experiment A probably should not be accepted when it comes at the cost of a 360× increase in metric learning time. In Experiment B, we see that LSML actually achieves a lower accuracy than the covariance metric, because it has optimised towards the classes available at training and not performed as well for the new test classes during classification (essentially overfitting), also leading to a significantly increased SVM training time. Therefore, we can avoid more computationally expensive metric learning by using a simple covariance metric transformation instead to achieve sufficiently good linear separability between identity classes. The use of the covariance metric as a feature descriptor has been established in the literature [13], but not necessarily in terms of colour histograms or as a superior alternative to more sophisticated metric learning. In general, this is a finding consistent with [17], which states briefly in their conclusion that "even a standard Mahalanobis metric not using any discriminative information yields quite good classification results". It is logically consistent that this effect is due to the metric learning algorithms generally overfitting to the training data, and the covariance metric remains relatively generalisable while being far less computationally expensive.

3.2.2. DukeMTMC4ReID Dataset

To further validate the results, we repeated the experiment on a much more challenging dataset, DukeMTMC4ReID [9], which is based on individuals detected with Doppia from the DukeMTMC dataset [11]. In this dataset, there are 1852 unique individuals (classes), with a total of 46,261 detections at varying bounding box resolutions. This is a more traditional person re-identification dataset in comparison to UoA-Indoor, with a small number of detections per class (as few as 10), making training much more difficult. Additionally, the dataset is captured in an outdoor setting, with eight cameras that are mostly not overlapping, with significant variation in pose and differing environmental conditions. In this dataset, the individuals are already cropped, which makes it harder to use DPM to consistently extract person parts without significant retraining. Instead, we use patch-based sampling to segment the cropped person images by dividing each person image into 16 parts in a regular 4x4 grid. While the image resolutions differ between detections (depending on if the person was closer or further away from the camera when they were detected) leading to differently sized parts, this does not affect our approach since the extracted features are normalised histograms for each part. The accuracy of this approach would likely be improved with more dense sampling [23], but for the purposes of this experiment, comparative measures are more important. The same colour and texture features are extracted as before - since there are now 16 parts per person instead of 8, the resultant feature vector is twice as long. Without dimensionality reduction, this would have significant impacts on the classification algorithm and its ability to deal with noisy data in more dimensions, but we use PCA to reduce the number of dimensions down to 32 to be consistent with the previous experiments.

Typically, a simple one-vs-all SVM would probably not be used in this scenario as there are too many different classes, leading to state space explosion and a high level of difficulty in separating the classes linearly. This makes following the SVM training time less reliable as an indicator of the impact of the different metric learning algorithms on separability, as the effect of the large number of classes dominates, as shown by the accuracy rates. However, we use the SVM here as a benchmark method for evaluating the different metric learning methods so that there is fair comparison with the previous experiments. We follow the same procedure as Experiment B, where not all classes are seen by the metric learning algorithm to determine the algorithm's ability to identify a generalisable metric for a given environment and emulate the real-world case. In this case, we provide 10% of the total classes to the metric learning algorithm (because there are far more classes in this dataset than UoA-Indoor), with the expectation that the learnt metric should be applicable to the other 90% of classes too. We also show the same metric learning methods as Experiment B, and used 5-fold cross validation to help with establishing more robust results since there is less data per class. It is important to note that the training times reported are for the entire dataset, not per detected person, so the computation times are naturally higher than in the case of UoA-Indoor since the dataset is larger.

As shown in Table 3, without metric learning the SVM essentially fails to learn. RCA failed with the larger dataset because of resource constraints. The increased number of classes and reduced number of samples per class causes the accuracy rate to be very low. The covariance metric still produces the best distance metric, with the best accuracy and the lowest metric learning time, while producing a similar SVM training time to the other methods. As before, LSML was very close to the covariance metric since it starts with the covariance metric as an initial guess and then converges towards a solution optimal for the given training set. Since we then apply the learnt transformation matrix to other previously unseen classes (in the same environmental settings), it seems that this causes the learnt matrix to move away from the more general solution.

The accuracy rate can be compared to the results reported in the original DukeMTMC4ReID [9] paper as rank 1 results. This has to be done carefully, as they use different features and a different evaluation protocol to the previous

Table 3: Experiment C: Metric Learning with a subset of classes, evaluated on 5-fold CV with SVM on unseen classes on DukeMTMC4ReID

Method	Accuracy (%)	Metric Learning Time (s)	SVM Training Time (s)
None	0.0896	0	323.25
LFDA	0.0899	0.048	326.37
LSML	22.28	11.43	365.95
Covariance	22.68	0.0096	375.77
RCA	N/A	N/A	N/A

Table 4: Comparison against DukeMTMC4ReID experiments from [9] (LOMO features)

Method	Accuracy (%)
KISSME	0.49
SVMML	9.61
PCCA	14.16
MFA	14.92
L2-norm	20.6
Covariance (from our exp)	22.68
RankSVM	23.19
XQDA	27.88
LSML (from our exp)	28.12
kMFA with exp kernel	38.29
kLFDA with exp kernel	38.56

experiments. Firstly, their work compares seven different types of features, with varying colour and texture choices. LOMO, which extracts HSV colour histograms and LBP texture features, is the most similar to the features used in the previous experiments. Secondly, their evaluation protocol selects 50% of the identities for the training set, rather than the 10% used in our previous experiments, so it is natural for the accuracy rates to be much higher generally since the training data represents a larger proportion of the global distribution. It is important to note that this is because [9] is only focusing on presenting baseline results for their new dataset, whereas in this paper the focus in on finding a metric learning algorithm that can learn a generalisable metric transformation with minimal training data, hence the difference in experimental design. To resolve this difference, for the purposes of comparison only, experiment C was re-run with only LSML and the covariance metric, with 50% of the samples provided to the metric learning algorithm. Naturally, since there is no actual learning in the covariance metric transformation, the accuracy here did not change when more data was allocated to training, which disadvantages it against the other algorithms presented that have more opportunity to learn a metric that suits the distribution of the dataset. Additionally, rather than mixing samples from all cameras for both testing and training, [9] fixes one camera as the probe, and detections for all other cameras are combined as the gallery. Importantly, no offline SVM is used in their experiments; instead, they simply pick the most similar source image for each probe and classify in that way. In Table 4, a rough comparison between some different metric learning algorithms on DukeMTMC4ReID is shown - other than the covariance metric and LSML, the results are obtained directly from [9].

The first interesting result to interpret is the increase in accuracy for LSML - from 22.28% under the previous evaluation protocol to 28.12%. This is an indication of the difference in accuracy that can be expected simply by changing how the accuracy is measured; in this case this is caused by the much larger amount of training data used. Secondly, the covariance metric result is still reasonably competitive against other, more complicated methods. In particular, the SVM-based methods SVMML and RankSVM are likely to require significant training time, and the kernel-based methods kMFA and kLFDA may not produce linearly separable classes that can be interpreted by a linear SVM. Unfortunately, no timing results are reported in [9], so it is not possible to identify the amount of speed that must be given up in exchange for higher accuracy rates. Given the focus on computationally light algorithms in this paper, that most intelligent applications require a more generalised metric that can be applied without training data that represents a large proportion of the global distribution, and that the covariance metric is not significantly worse than most of the other state-of-the-art methods, the covariance metric is perhaps the best option for making the feature vectors easier to classify in the proposed re-identification scheme. This is an important result, as other researchers have continued to develop and use more sophisticated metric learning approaches for person re-identification.

In terms of future work, there are other metric learning approaches that could be investigated. Newer methods such as Marginal Fisher Analysis (MFA), Pairwise Constrained Component Analysis (PCCA) [24], and Cross-view Quadratic Discriminant Analysis (XQDA) [25] may perform better, but there is no indication that they are able to overcome the overfitting problem suffered by the existing metric learning algorithms; [26] alludes to MFA requiring all subject identities during training in order to yield a good metric, and the data from DukeMTMC4ReID in Table 4 would indicate poorer performance even with 50% of the identities made available during training. Deep transfer metric learning [20] allows for a partially trained model to incorporate new distributions from new datasets quickly to learn a more specific metric for a new environment. X^2-LMNN metric learning [27] has been shown to be particularly suitable for histogram-based data. However, both methods are all likely to be more computationally expensive than the covariance metric while still overfitting, and the covariance metric already yields sufficiently good results.

4. Classification

Once a detected person has been passed through feature extraction, dimensionality reduction, and metric learning, the resultant feature vectors can then be classified. The difficulty lies in the fact that not only is there significant variation in the appearance of a person between camera views, but even within the same camera view environmental factors such as lighting can have an impact. Within-class variation makes classification challenging, especially if there are overlaps between different classes. An example of this is when people are standing directly underneath a bright light - depending on the angle of the camera, this can result in the appearance of the person being saturated (i.e. a lot of the pixels are at their maximum value of 255, representing white), making it difficult to visually distinguish between two different people. The aim of the classification step is to attempt to overcome these issues and correctly identify the individual (where each individual is represented as a class) based on a feature vector representing their appearance. Using classification is distinct to approaches that learn a transfer function between each pair of camera views in order to model the environmental and parameter changes [19].

When selecting the classification algorithm, a number of practical issues need to be considered. Since the vast majority of modern approaches use deep learning, the availability of training data is critical. However, in many intelligent applications, the identities are unconstrained. For example, in a supermarket, it is not possible to collect samples of all possible customers before they enter the store. Without available training data, then our options are limited to those that can successfully use transfer learning, or unsupervised methods that can learn dynamically without any training. Purely unsupervised methods are difficult to use successfully, and often have low accuracy rates because they can irrecoverably diverge from the correct solution. Instead, a compromise can be found in one-shot learning methods, which use a single sample during training to essentially initialise the model. Having even one sample can significantly help improve the accuracy of classification during runtime, especially if the system continues to learn from the streaming data.

Another practical issue is the use of low-cost computing environments in intelligent applications such as embedded systems that have limited computation power and memory. Many modern "real-time" solutions in the literature are implemented on GPUs or cloud computing resources, which may be impractical for the real-world. Therefore, computationally efficient approaches are preferred, especially those that are scalable to support many identity classes. This means that methods that suffer from state space explosion such as decision trees are unsuitable. In this work, we tested four algorithms that use one-shot learning, with two of those developed by us and previously presented in our conference paper [8]. The previous experiments have already shown that the UoA-Indoor data is mostly linearly separable after dimensionality reduction and calculating the covariance metric, so it can be easily shown that most of the errors during classification are attributable to the algorithm, rather than the data.

4.1. Algorithms

4.1.1. Support Vector Machine and Perceptron

The linear Support Vector Machine (SVM) is a well-understood machine learning method for classification with two classes. The SVM iteratively forms a hyperplane in multi-dimensional space between the classes so that the distance between the hyperplane and the data on each side is as large as possible. SVMs can then be used in a multi-class context in a one-against-all scheme, where one SVM is trained for each class, and the hyperplane separates between the target class and all other classes. When testing data is presented, it is evaluated against the models for each class, and the most positive model (i.e. the model where the input data is the furtherest away from the hyperplane on the correct side) is determined the winner. Other approaches such as RankSVM [28] can also be used for this task, including the use of the kernel trick to learn in a non-linear way in high-dimensional space, but these are not necessary as the use of metric learning has already transformed the data to be linearly separable. Since the data in our case is relatively sparse (i.e. the number of dimensions is low relative to the number of samples), the basic SVM formulation is more appropriate, especially considering that it has a significantly lower computation time during execution than the most complex models. The SVM is configured to use one-shot learning by providing a single sample from each class initially, and then using online learning as the data is streamed to the SVMs.

A simple Perceptron was also used to provide an alternative benchmark to the SVM. It can be difficult to form good separation functions with a single multi-class perceptron, especially where there is significant noise or variation in the inputs, so a one-against-all scheme is used to match the SVM model. A multi-layer perceptron (MLP) network was also tested, but the accuracy rate was similar to the perceptron at a much higher computation cost, so it was rejected. The use of deep neural networks were similarly rejected, since even though they could reach higher accuracy rates, the high computation costs would not be justified in the target intelligent application. The perceptron was similar configured as the SVM to use one-shot learning, with an anchor sample for each class to initialise each perceptron.

Both the SVM and Perceptron are likely to suffer in a one-shot setting, because even a single misclassification can have a long-term effect. The repeated online training epochs essentially encode the error and the models are not very robust against noise, even if we use early stopping to avoid overfitting. Therefore, some alternative one-shot learning methods are needed.

4.1.2. Gallery

The Gallery algorithm essentially maintains a model containing multiple samples for each class, so that there are many different examples of the same person. The two main steps, classification and model update, are based on the ViBE [29] algorithm, which is used in background estimation for classifying pixels as being part of the background or foreground. It has been shown to be relatively robust against noise, and phases out unwanted noise samples over time. Adapting ViBE for person re-identification meant removing the unnecessary parts of the algorithm, such as the propagation of samples between neighbouring models.

Each class is modelled by a gallery of N samples (*targets*). When a person is seen for the first time, a new class needs to be initialised. The extracted feature vector is used to anchor the model for that class, and all N samples are initially identical. For each person detected, the new feature vector (*source*) is taken as an input into the person re-identification module, and is compared to each target sample in each class model. The comparison is evaluated using the Euclidean distance between the source and each target sample, since all of the feature vectors have been transformed to be in Euclidean space. A lower distance means that the two vectors are more similar; a parameter DIS is set to indicate when a distance is low enough to be considered a match to the target class. However, there may be more than one target sample that is close enough to the source, and these may be in multiple target classes. The rule that is used from ViBE is that for each class, if the number of matches is above a minimum number ($numMin$), then the source feature vector is classified as part of that target class. The effect is that the first class encountered that has enough samples is determined the winning class, reducing the computation time by eliminating unnecessary comparisons when the matching class has already been found. There are a number of parameters that can have a significant impact on the accuracy, so the best values were found empirically through a parameter sweep, with N = 40, DIS = 40, and $numMin$ = 5. In other words, the source sample needs to match at least 5 out of the 40 samples in a gallery class, where the two feature vectors need to have a distance less than 40 (which is a unitless value) to be declared a match. This classification scheme can be described formally, using $\mathbf{x}$ as the source feature vector and $\mathbf{s}$ as the target feature vector(s), with:

$$\mathbf{x} \in c \; if \; C(\mathbf{s}) >= numMin \tag{2}$$

c is the class number and $C(\mathbf{s})$ is the number of feature vectors $s_0, s_1, ...s_N$ in classs c that satisfy the condition:

$$\| \mathbf{s} - \mathbf{x} \| < DIS \tag{3}$$

where $\|\mathbf{s} - \mathbf{x}\|$ is essentially the Euclidean distance between the target and source vectors. Once the matching class has been identified, the model is updated to incorporate the new sample into the model. A random sample in the class gallery is replaced with the source, with each target sample having a uniformly equal chance of replacement. While this random behaviour is non-deterministic, statistically this causes each sample in the model to have a smooth decaying lifespan (i.e. inverse probability of being replaced), so that the gallery to be comprised of both more recent and older samples. This helps improve the robustness of the model by reducing the impact of single misclassifications since *numMin* matches need to be made, while also allowing for different viewpoints, lighting conditions, and other factors that can significantly change the appearance of a person. Constraining the size of the gallery to N samples helps ensure that the method is scalable and does not grow over time.

4.1.3. Sequential k-Means

The last algorithm developed and tested was Sequential k-Means, which uses a modified form of k-means clustering inspired by [30] and modified to fit this re-identification context. The underlying objective is to use k-means clustering with online learning, where each cluster represents a new identity class. When a person appears for the first time, that first sample is used to intialise a new cluster center with that feature vector directly. For each subsequent source feature vector, it is compared to the cluster means. Since the covariance metric transformation is applied, determining the closest cluster mean to any input feature vector is as simple as calculating the Euclidean distance, where a lower distance indicates a more similar vector, formally described as:

$$\| \mathbf{m}_c - \mathbf{x} \| \tag{4}$$

where $\mathbf{m}_c$ is the cluster mean for class c and $\mathbf{x}$ is the source input feature vector. Apart from computational efficiency due to the lightness of the distance calculation, this method also has relatively low memory requirements, as each class is only represented by a cluster mean; unlike normal k-means clustering, each data point is not collected/stored. This is achieved by updating the winning class/cluster mean using:

$$\mathbf{m}_c = \beta \mathbf{x} + (1 - \beta)\mathbf{m}_c \tag{5}$$

where β is a constant weighting factor that determines how quickly the cluster mean adapts to changes. This is a form of linear filtering, where each new sample can contribute to the mean while still allowing past samples to have an influence, depending on the value of β. Using the UoA-Indoor dataset and a parameter sweep, $\beta = 0.1$ was found as the value where the accuracy results were the best, although this may be dataset dependent as the rate of change in visual appearance is likely to impact how quickly the model should adapt. Choosing a good value for β depends on balancing the need to update the model so that it is more similar to future samples in the short-term with the desire to keep the model robust against noise and misclassification. It should also be noted that the use of k-means clustering generally may have a potentially negative effect of creating spherical clusters and decision boundaries, in comparison to most of the other tested methods which have linear boundaries.

4.2. Experimental Settings

In order to compare the four algorithms, each method was run using the transformed feature vectors are PCA (32-dimensions) and metric learning (covariance metric transformation). The feature vectors are streamed in order of appearance in the recorded footage, with all four camera views mixed together. The first sample of each class is labelled, which is provided to the algorithm as an anchor for the new identity class. All other vectors are labelled by the algorithm, and also used for updating the internal models. Accuracy and computation times are averaged across five trials; since there is only one sample used for "training", there are 19 samples in the training set and over 28,000 unlabelled samples in the test set. For the SVM and perceptron methods, the number of output classes has to

Table 5: Classification Accuracy and Computation Time on UoA-Indoor

Method	Accuracy (%)	Classification Time (s)	Classification Time (kfps)
Sequential k-Means	67.22	0.282	99.3
Gallery	48.41	4.720	5.9
One-shot Perceptron	34.44	29.797	0.94
One-shot SVM	29.61	28.891	0.97

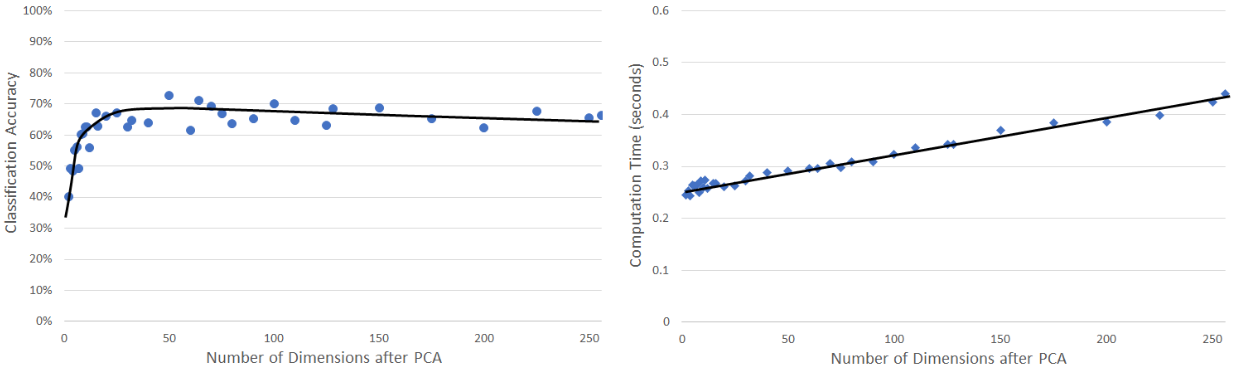

Figure 6: Classification accuracy (averaged over five tests) and computation time (per test) from a Sequential k-Means method classifying approx. 28,000 feature vectors across 19 classes with between 2 and 256 dimensions after PCA and a covariance metric transformation.

be defined at initialisation since the right number of models need to be created for training one-against-all models, whereas Gallery and Sequential k-Means are more flexible and can dynamically increase the number of classes as needed. The ideal result would be to find an algorithm that has high classification accuracy and low computation time (low in terms of seconds, high in terms of frames per second). All accuracies and computation times are derived from the same computer as the metric learning experiments, with the algorithms written in Python with the use of Numpy and Scikit-learn. The classification times should be compared to each other on this machine, rather than taken as evidence for real-time execution generally. Note that the time given in seconds is the time required to process the entire dataset, and the second time given in kfps shows how many thousands of frames can be processed per second.

4.3. Results

Keeping in mind that the offline SVM trained in Experiment B reached 98.82% accuracy, the results for the one-shot learning approaches are shown in Table 5. There is an appreciable drop in accuracy, which is to be expected since far fewer samples are used for training. For the SVM and Perceptron, not only is the accuracy rate very low, but this causes the computation times to significantly increase as well as the high error rates cause the algorithm to attempt to keep learning instead of reaching an error tolerance level. The Gallery method still has a relatively high classification time due to the need to iterate through many target samples in the gallery for each possible class, and has a middling accuracy rate as the samples may not be replaced quickly enough to stay up-to-date. It is clear that Sequential k-Means is the winning method, with both the highest accuracy rate and the lowest computation time for classification. However, losing 30% accuracy between the supervised and one-shot learning methods needs to be further mitigated in order for these algorithms to be viable for real-world applications. We may not be able to sufficiently overcome inter-camera variation when mixing the camera views together with these one-shot learning methods.

4.4. Tuning Sequential k-Means

For the purposes of fair comparison, the same data was used for all four algorithms, but the parameters used in previous steps can be tuned to improve the overall accuracy of the system. Firstly, the number of dimensions to

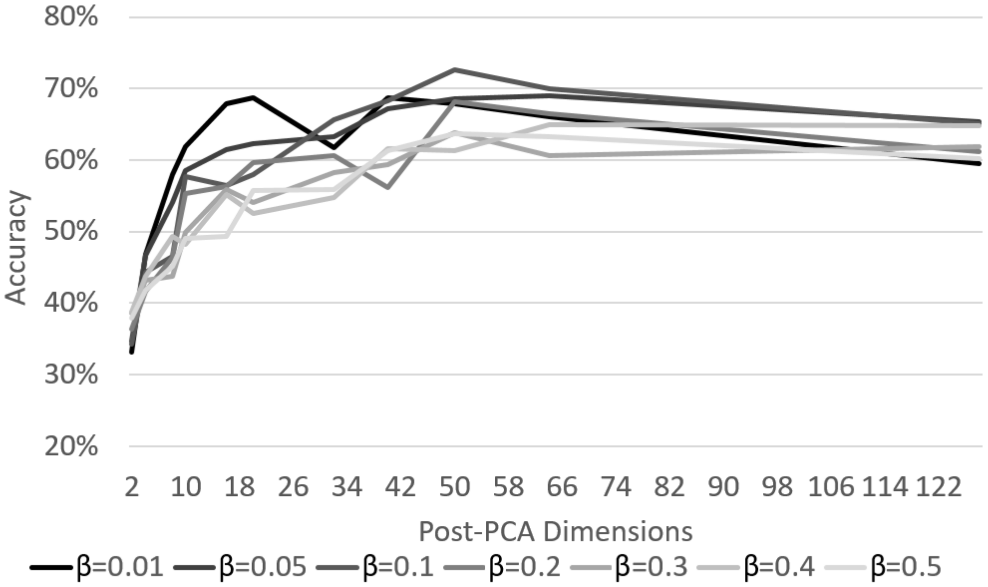

Figure 7: Classification accuracy (vertical axis) from Sequential k-Means depending on the number of dimensions after PCA (x-axis) for different model update rates β. Lighter colours represent higher model update rates.

include in the feature vectors after PCA was re-tested, since the original experiments used an offline SVM. Figure 6 shows an accuracy curve that is very similar to the one in Figure 5. However, the computation time for Sequential k-Means increases linearly (rather than exponentially for the SVM), since the model size linearly increases as the feature vectors become longer. Selecting 50 dimensions for the PCA reduction improved the accuracy of Sequential k-Means from 67.22% to 72.65%, a substantial improvement of over 5% just from increasing the number of dimensions from 32 to 50.

Secondly, the update rate β was investigated for different post-PCA feature vector lengths, as the 0.1 value was determined based on 32 dimensions as well. However, it appears that these two parameters are largely independent, i.e. changing the number of dimensions has negligible effect on the impact of changing β on accuracy. This can be seen in Figure 7, where the main shape of the curve is roughly the same regardless of the model update rate, and as the model update rate increases the accuracy decreases. This implies that the two parameters likely have independently additive effects. The highest accuracy is achieved when β is between 0.05 and 0.1, around 50-dimensions post-PCA. Variations in β have no impact on computation time since it is just a constant used in a multiplication operation. Further improvements to our Sequential k-Means formulation include introducing a ranking and re-ranking scheme, which would allow the algorithm to take the Euclidean distances to all cluster means into account. However, this would increase the computation time significantly for marginal improvements in accuracy, whereas incorporating other types of features that carry different types of discriminating information may be more beneficial.

5. Conclusions

For many intelligent applications, having a good understanding of who is in what places and what time would be of great help. However, person re-identification between multiple camera views remains a challenging problem, especially when real-world practical issues are taken into account. This work focuses on two parts of the re-identification challenge, metric learning and classification. We establish that the covariance metric transformation is sufficient for re-identification applications; this could be further established on other datasets to determine the wider generalisability of this claim. The primary reason for the covariance metric transformation's superior performance is due to the more sophisticated metric learning algorithms generally overfitting to the training data, failing to remain generalisable enough for new unseen test data. In particular, this claim may be more applicable to real-world re-identification scenarios, where there are a large number of samples per class as we observe them over time, and a smaller number of classes visible over any reasonable time period, rather than the standard dataset formulation of a large number of classes with a very small number of gallery images per class. However, we have demonstrated that it is still competitive against more sophisticated metric learning approaches on a large standard person re-identification dataset. We also evaluate four one-shot learning methods for classifying the processed feature vectors into identity classes, and show that our Sequential k-Means approach outperforms other computationally efficient methods in terms of both accuracy and computation time. Importantly, this relies on the assumption that the model can continue to update and develop over time, in contrast to the supervised learning approaches developed on most existing re-identification datasets. We also discuss parameter selection and scalability for the Sequential k-Means method, and improve the accuracy of our algorithm to 72.65%.

References

[1] X. Wang, Intelligent multi-camera video surveillance: A review, Pattern Recognition Letters 34 (2013) 3–19.

[2] A. Dick, A. v. d. Hengel, H. Detmold, Large-Scale Camera Topology Mapping: Application to Re-identification, Springer London, 2014, pp. 391–411.

[3] E. Ahmed, M. Jones, T. K. Marks, An improved deep learning architecture for person re-identification, in: Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 3908–3916.

[4] T. Xiao, S. Li, B. Wang, L. Lin, X. Wang, Joint detection and identification feature learning for person search, in: Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 3415–3424.

[5] O. Camps, M. Gou, T. Hebble, S. Karanam, O. Lehmann, Y. Li, R. J. Radke, Z. Wu, F. Xiong, From the lab to the real world: Re-identification in an airport camera network, IEEE Transactions on Circuits and Systems for Video Technology 27 (2017) 540–553.

[6] I. Filković, Z. Kalafatić, T. Hrkać, Deep metric learning for person re-identification and de-identification, in: International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2016, pp. 1360–1364.

[7] A. T.-Y. Chen, M. Biglari-Abhari, K. I.-K. Wang, A computationally efficient pipeline for camera-based indoor person tracking, in: Image and Vision Computing New Zealand (IVCNZ), 2017.

[8] A. T.-Y. Chen, M. Biglari-Abhari, K. I.-K. Wang, Fast one-shot learning for identity classification in person re-identification, in: IEEE International Conference on Control, Automation, Robotics and Vision (ICARCV), 2018.

[9] M. Gou, S. Karanam, W. Liu, O. Camps, R. J. Radke, Dukemtmc4reid: A large-scale multi-camera person re-identification dataset, in: Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2017, pp. 10–19.

[10] A. Bedagkar-Gala, S. K. Shah, A survey of approaches and trends in person re-identification, Image and Vision Computing 32 (2014) 270–286.

[11] E. Ristani, F. Solera, R. Zou, R. Cucchiara, C. Tomasi, Performance measures and a data set for multi-target, multi-camera tracking, in: European Conference on Computer Vision (ECCV) Workshop on Benchmarking Multi-Target Tracking (BMTT), 2016, pp. 17–35.

[12] U. Park, A. K. Jain, I. Kitahara, K. Kogure, N. Hagita, Vise: Visual search engine using multiple networked cameras, in: International Conference on Pattern Recognition (ICPR), 2006, pp. 1204–1207.

[13] S. Bak, E. Corvee, F. Bremond, M. Thonnat, Person re-identification using spatial covariance regions of human body parts, in: International Conference on Advanced Video and Signal Based Surveillance (AVSS), 2010, pp. 435–440.

[14] K. W. Lee, N. Sankaran, S. Setlur, N. Napp, V. Govindaraju, Wardrobe model for long term re-identification and appearance prediction, in: International Conference on Advanced Video and Signal-based Surveillance (AVSS), 2018.

[15] G. Shu, A. Dehghan, O. Oreifej, E. Hand, M. Shah, Part-based multiple-person tracking with partial occlusion handling, in: Conference on Computer Vision and Pattern Recognition (CVPR), 2012, pp. 1815–1821.

[16] I. Kviatkovsky, A. Adam, E. Rivlin, Color invariants for person reidentification, IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 35 (2013) 1622–1634.

[17] P. M. Roth, M. Hirzer, M. Köstinger, C. Beleznai, H. Bischof, Mahalanobis Distance Learning for Person Re-identification, Springer London, London, 2014, pp. 247–267.

[18] A. Bellet, A. Habrard, M. Sebban, A survey on metric learning for feature vectors and structured data, CoRR abs/1306.6709 (2013).

[19] T. Avraham, I. Gurvich, M. Lindenbaum, S. Markovitch, Learning implicit transfer for person re-identification, in: European Conference on Computer Vision Workshops (ECCVW), 2012, pp. 381–390.

[20] J. Hu, J. Lu, Y. P. Tan, J. Zhou, Deep transfer metric learning, IEEE Transactions on Image Processing (TIP) 25 (2016) 5576–5588.

[21] A. T.-Y. Chen, The Computers Have a Thousand Eyes: Towards a Practical and Ethical Video Analytics System for Person Tracking, Ph.D. thesis, 2019.

[22] F. Xiong, M. Gou, O. Camps, M. Sznaier, Person re-identification using kernel-based metric learning methods, in: European Conference on Computer Vision (ECCV), 2014, pp. 1–16.

[23] R. Zhao, W. Ouyang, X. Wang, Person re-identification by salience matching, in: International Conference on Computer Vision (ICCV), 2013, pp. 2528–2535.

[24] A. Mignon, F. Jurie, PCCA: A new approach for distance learning from sparse pairwise constraints, in: Conference on Computer Vision and Pattern Recognition (CVPR), IEEE Computer Society, 2012, pp. 2666–2672.

[25] S. Liao, Y. Hu, X. Zhu, S. Z. Li, Person re-identification by local maximal occurrence representation and metric learning, in: Conference on Computer Vision and Pattern Recognition (CVPR), IEEE Computer Society, 2015, pp. 2197–2206.

[26] M. Faraki, M. T. Harandi, F. Porikli, No fuss metric learning, a hilbert space scenario, Pattern Recognition Letters 98 (2017) 83–89.

[27] D. Kedem, S. Tyree, F. Sha, G. R. Lanckriet, K. Q. Weinberger, Non-linear metric learning, in: Advances in Neural Information Processing Systems (NIPS), 2012, pp. 2573–2581.

[28] D. Gray, H. Tao, Viewpoint invariant pedestrian recognition with an ensemble of localized features, in: European Conference on Computer Vision (ECCV), 2008, pp. 262–275.

[29] O. Barnich, M. Van Droogenbroeck, ViBe: A universal background subtraction algorithm for video sequences, IEEE Transactions on Image Processing (TIP) 20 (2011) 1709–1724.

[30] R. Duda, Sequential k-means clustering, 2008.

Andrew Tzer-Yeu Chen is currently pursuing a Ph.D. degree in Computer Systems Engineering at The University of Auckland. He received the BE(Hons) degree in Computer Systems Engineering with First Class Honours and the BCom degree in Marketing, and Innovation and Entrepreneurship from the University of Auckland in 2015. His research interests include computer vision, hardware/software co-design, machine learning, robotics, and engineering education.

Morteza Biglari-Abhari is a Senior Lecturer with the Department of Electrical and Computer Engineering at The University of Auckland. He received the M.Sc. degree in Electrical and Electronic Engineering from Sharif University of Technology, and the Ph.D. degree from the University of Adelaide. His research interests include computer architecture, hardware/software co-design of real-time and secure embedded systems, and energy-efficient multiprocessor systems on chip.

Kevin I-Kai Wang is a Senior Lecturer with the Department of Electrical and Computer Engineering at The University of Auckland. He received the Ph.D. degree in Electrical and Electronics Engineering from the University of Auckland in 2009. His current research interests include distributed computational intelligence, pervasive healthcare systems, industrial monitoring and automation systems, and bio-cybernetic systems.