Pattern Recognition Coursework 2

Paul Streli - ps4715
Imperial College London
CID: 01103106

Karoly Horvath - kth15
Imperial College London
CID: 01088730

Abstract

This report describes person re-ID experiments on the popular CUHK03 dataset that contains pictures of pedestrians taken from two different surveillance cameras. A set of features was already provided. The goal is to find a suitable distance metric which learns a feature transformation yielding improved performance on a range of metrics for kNN-retrieval. The original covariance based Mahalanobis method, Large Margin Nearest Neighbour Distance Metric, Metric Learning for Kernel Regression and a fully connected Neural Network with Triplet Loss are considered for this purpose. The Kernel Regression gave the best results with a performance similar to the untransformed baseline approach.

1. Introduction

Information retrieval is the process of returning a certain n number of relevant items from a set, given a query item. The relevance between the query item and the returned items are defined by the retrieval problem and the method of selecting the n most relevant items is usually the point of interest. [8]

An example could consist of a set of pictures of different fruits (apples, pears, peaches, etc.) and an image of an apple as the query item. [9] The problem is defined as returning all of the images of apples from the set. Here, the relevance is defined as the subject of the picture, and the problem can be tackled in different ways.

In this report, a similar scenario will be considered, and the different ways of implementing image retrieval systems will be considered. 1360 pedestrians were photographed multiple times and the set of pictures taken are given. The query item is a picture of one of the pedestrians taken with a camera, and the objective is to return n images from the set that were taken of the same person by the other camera. For this task several distance metric learning approaches will be investigated.

1.1. Data Sets

Three sets of images are given, the so called training set, the gallery set and the query set. The images are from the CUHK03 data set [4] and the extracted features are also given, therefore the investigation will exclude feature creation.

The items in the query set are used to query the gallery set. The system shall return the images of the same person from the gallery set. In general, the gallery and the query set both consist of pictures taken from the pedestrian using both of the cameras. To keep the real-life point of the exercise described in 1, the gallery set should only contain pictures of the same query identity taken from the other camera. To achieve this, before performing the information retrieval process, the unnecessary pictures are removed from the gallery set. These sets can be thought of as the test set, which are used to evaluate the distance metrics that are learnt according to the target functions described in 2.

The training set is used for validating and training the models. To reproduce the original information retrieval problem in the validation set, it is constructed by taking all of the samples of 100 randomly selected pedestrians from the training set. Furthermore, it needs to be split into a validation query and a validation gallery set.

1.2. Feature Extraction

The three sets of features initially provided, the training, gallery and query sets were created from the CUHK03 data set [4], which consists of 13160 images of pedestrians. For extracting the given features, the set was divided into a training and a testing subset. The training samples were repeatedly fed into a residual network, using a loss function that separates the data points with different labels as far as possible while keeping the same labels as close as possible. After optimising the feature vectors to this criterion, the resulting 2048 features were provided as the training set for the coursework. The test (gallery and query) images were fed into the same residual network to map the corresponding feature vectors to the same space as the training set. As the network was optimised for the training set, the training set is very well clustered. However, the test set will be more scattered due to the loss of generality.

2. Evaluation Methods

The training set is used to find a suitable model, which is a space transformation that ideally separates the test pictures with different labels spatially in the transformed space while keeping the ones with same labels close to each other. After having transformed the gallery and the query sets to the learnt optimal subspace, the evaluation of the model is achieved using the k-nearest-neighbours method (kNN). For every query image, the k closest neighbours are returned using Euclidean distances in the transformed space. The labels of the returned neighbours

are examined and the Precision, Recall, Mean Average Precision (mAP) and rank@N metrics are calculated together with the CMC Curve (see Appendix A). To evaluate the investigated models, the mean average precision metrics are compared and quoted in this report.

3. Formulation of the Objective

The mathematical description of the machine learning problem formulated in the previous section, is given by the following objective function. A metric is learnt that brings similar data points closer, that is it is desired to minimise the distance between feature vectors that are of the same label in the gallery set:

$$\min_{T} \sum_{i \neq j} d[T(x_i), T(x_j)] \tag{1}$$

where x_i and x_j are a pair of samples corresponding to the same person, d signifies the Euclidean distance between the two arguments given to it and T is the transformation (model) applied to the features.

At the same time, it is desired to push the dissimilar data points further apart from each other, that is to maximize the distance between the feature vectors of different pedestrian identities.

$$\max_{T} \sum_{i,k} d[T(x_i), T(x_k)] \tag{2}$$

where d and T are defined as in Equation (1) and x_i and x_k are a pair of samples belonging to different pedestrians in the gallery set.

4. Baseline Approaches

As a baseline approach, the evaluation experiment will be performed without applying any transformations to the original features. These baseline metrics serve comparison purposes, to measure whether the more advanced models serve as a more adequate solution to the retrieval problem. This approach simply considers the k closest gallery points in the original 2048 feature space to each of the query images when calculating the evaluation metrics. This approach yielded an mAP of 0.4353.

Alongside using k-nearest-neighbour evaluation metrics, kmeans clustering was also incorporated as an additional baseline evaluation method. In this method, the transformed test set is divided into k number of clusters based on how close the data points are to each other. The cluster boundaries are learnt and the predicted label of the query pictures are determined based on which boundary they belong to. The proportion of the query images that were correctly identified will be quoted as kmeans accuracy and the comparison will be performed for the trained models. Without applying any transformations to the features the calculated kmeans accuracy is 0.7086.

It is important to note that the cluster boundaries should be learnt on the gallery set with the unnecessary images removed. Since this means that the gallery set is different for each query image, the kmeans cluster boundaries would need to be calculated 1400 times (the number of query images). An important consideration in machine learning problems is the time taken to reach a conclusion. It was decided that it is too computationally expensive to perform kmeans clustering and calculate the kmeans accuracy for all of the different models using the previously mentioned method. However, calculating the cluster boundaries on the entire gallery set and removing the unnecessary data points before computing the evaluation metrics results leads to only slightly modified cluster boundaries with respect to the desired ones. This will be suitably close enough to the desired boundaries, such that the kmeans accuracies calculated with this method are good enough to compare the different models. This decreases the computation time by a factor of 1400.

5. Distance Metric Learning

During distance metric learning, the training set is used to learn an optimal transformation that separates the data points with different labels spatially while keeping the same labels close to each other. The learning should generalise well to the test set. In each subsection the challenges that each of the proposed approaches addresses and the way they are solved are described.

Due to the highly clustered nature of the training features, it is expected that using distance metric learning, it will be difficult to learn a separative transformation that achieves better mean average precision than the baseline method.

5.1. Validation

To select the optimal parameters for the distance metric learning models, a validation set is formed from the training set as described in Section 1.1. This set, which consists of the validation gallery and query sets is therefore also highly spatially clustered according to the person IDs. Inherently this results in a very high mean average precision reaching up to 1 - even for the baseline method when applied on the validation query and gallery sets. This makes the given validation set unusable for proper parameter selection. Thus, it was decided to use the small changes in the validation error for hyperparameter optimisation in conjunction with deviations in the actual test error - determined on the actual gallery and query set. While this approach does not conform with the basic conventions of machine learning as it can lead to misleading results in performance and a loss in generalisation, it seems to be the best approach for the given conditions.

5.2. Feature Reduction

Given the high number of features, it appears reasonable to perform feature reduction. A smaller set of features might lead to shorter computation times and generalise better to unseen data. Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) were discussed in the previous coursework and can be applied in this context [7]. As can be seen from Appendix B, most of the variance in the feature space can be explained by the first 100 principal components. The principal components corresponding to the 500 largest eigenvalues already explain more than 99% of the total data variance. As expected, the information loss from feature reduction

using PCA is insignificant (mAP with 500 principal components = 0.4357, mAP with 100 principal components = 0.4350). While LDA is a supervised method, PCA usually performs better than LDA on non-multivariate Gaussian distributions [11]. This was also the case for the given distribution as mAP did decrease after feature reduction using LDA (mAP with LDA-500 features = 0.3745, mAP with LDA-100 features = 0.3618). This trend could be observed when LDA and PCA were used in conjunction with more advanced distance metrics method as well. The reason for this might be the pre-trained feature space that already separates the samples in an almost optimal way. PCA is better at capturing the relative spacing of the original distribution. In the following, PCA will be used for feature reduction.

5.3. Mahalanobis distance learning

Mahalanobis distance learning is a well-studied approach for learning a feature transform that reduces the Euclidean distance between related samples. In its general form, that was discussed in lectures, the distance between the untransformed points $\mathbf{x_1}$ and $\mathbf{x_2}$ is given by $d_A(\mathbf{x_1}, \mathbf{x_2}) = (\mathbf{x_1} - \mathbf{x_2})^T A(\mathbf{x_1} - \mathbf{x_2})$ where the Mahalanobis matrix A can be decomposed as $A = G^T G$ by Cholesky decomposition. The transformed feature vector $\mathbf{x_1^*}$ is thus given by $\mathbf{x_1^*} = G\mathbf{x_1}$.

For the implementation of the Mahalanobis distance learning algorithms, the Python *metric-learn* library was used [2].

5.3.1 Original Mahalanobis distance

As a baseline for Mahalanobis feature transformation, it was decided to start with the original covariance based Mahalanobis approach that was introduced in 1936 [5]. The Mahalanobis Matrix is the inverse of the features covariance matrix computed using the equation $A = (\frac{1}{N-1} X X^T)^{-1}$, where X is the matrix composed from the training samples in its columns all centered by the corresponding feature means and N is the number of samples. The resulting Mahalanobis distance takes into account the scaling of the features and the expected distribution around a set's centroid. Since the same features are used for training and testing their covariance matrices can be expected to be similar given they are sufficiently large and drawn from the same distribution. Thus, the original Mahalanobis matrix was calculated on the whole training set and then used to transform the query and gallery features. Performing the retrieval experiments on the transformed features, the following results were obtained: Applying the covariance based method directly on the original training set led to a drastically drop in mAP (0.2337). The performance of the retrieval experiment could be improved by reducing the feature space to 50 dimensions with PCA (mAP = 0.3902) indicating that the feature reduction acts stabilising and filters out the noisy components that are amplified by the computation of the inverse covariance matrix. While changes were small with mAP varying from 0.9704 (2048 dimensions) to 0.9993 (50 dimensions), during the validation process it also became apparent that feature reduction is beneficial. Overall, the original Mahalanobis method did not lead to improvements compared to the baseline. This is not unexpected given the unsupervised nature of the Original Mahalanobis Distance and the fact that the original feature space was already pre-trained so that similar IDs are spatially close to each other.

5.3.2 Large Margin Nearest Neighbour

As next step, it was decided to take advantage of a supervised method. The Large Margin Nearest Neighbour was designed to ensure that the k-Nearest-Neighbours for each sample belong to the same ID while samples belonging to a different class are further away [10], aligning with the formulation of the Machine Learning problem described in Section 3. For this purpose the following loss function was defined for the transformation matrix G:

$$\epsilon(G) = \sum_{ij} \eta_{ij} ||G(\mathbf{x_i} - \mathbf{x_j})||^2 + c \sum_{ijl} \eta_{ij} (1 - y_{il}) \max(1 + ||G(\mathbf{x_i} - \mathbf{x_j})||^2 - ||G(\mathbf{x_i} - \mathbf{x_l})||^2, 0) \quad (3)$$

c is a positive constant, η_{ij} indicates whether $\mathbf{x_i}$ and $\mathbf{x_j}$ are target neighbours (usually defined as $\mathbf{x_i}$'s k nearest neighbours with the same ID) and y_{il} is equal to 1 when $\mathbf{x_i}$ and $\mathbf{x_l}$ share the same ID and 0 otherwise. The corresponding formulated optimisation problem can be found in Appendix C. This approach will yield a solution that will only penalise the separation between local samples (i.e. target neighbours). This also allows the formation of several clusters for a single ID and thus is well-suited for a kNN classification problem. As the untransformed gallery and training features seem to be distributed into clusters, the algorithm appears as a good choice for our kNN retrieval problem given a well-chosen hyperparameter k.

Indeed, the obtained results were better compared to the baseline Mahalanobis metric achieving an mAP of 0.4047 on the original feature set and $k = 3$. While the computation time and the memory usage significantly decreased for the LMNN metric, the retrieval error stayed almost constant when the feature space was reduced to 500 dimensions (according to marginal variations in validation error) beforehand, underlining the strong benefits of feature reduction using PCA. As the validation error was very close to 0 (mAP = 0.998), it was difficult to select the optimal value for k. Only after evaluating on the test set, it became clear that the optimal settings for this distance metric are a PCA-reduced feature space of 500 dimensions and $k = 5$ (equal to minimum number of samples per ID in training set) - mAP = 0.4112.

5.3.3 Metric Learning for Kernel Regression

Distance metric learning can be also used in conjunction with Kernels. The Metric Learning for Kernel Regression (MLKR) method is based on kernel regression where the label of a sample $\mathbf{x_i}$ is approximated by a weighted average as follows $\hat{y}_i = \frac{\sum_{i \neq l} y_j k_{ij}}{\sum_{i \neq l} k_{ij}}$. y_i is the real ID of

$\mathbf{x_i}$ and k_{ij} is a Gaussian kernel function given by $k_{ij} = \frac{1}{\sigma\sqrt{2\pi}} \exp - \frac{(\mathbf{x_1}-\mathbf{x_2})^T A(\mathbf{x_1}-\mathbf{x_2})}{\sigma^2}$ (Note that $A = G^T G$ is a Mahalanobis matrix). The loss function is then defined as $\epsilon = \sum_i (y_i - \hat{y}_i)^2$ and the gradient becomes $\frac{\delta L}{\delta G} = 4A \sum_i (y_i - \hat{y}_i) \sum_j (y_i - \hat{y}_j) k_{ij} (\mathbf{x_1} - \mathbf{x_2})(\mathbf{x_1} - \mathbf{x_2})^T$. According to [12] this method can be seen as a scaling-invariant supervised version of PCA. Moreover, due to the design of the loss function the transformed distances better reflect the similarity in ID. For training, G was initialised with the principal components of the desired dimension.

Validation could not be utilised for this method as the mAP value on the validation set was always equal to one. Training the MLKR model in the original feature space without MLKR feature reduction led to an mAP of 0.4354. Prior reduction of the input features to 500 dimensions led to a mAP even slightly better than baseline (0.4356) and a reduction in computation time. Further feature reduction using PCA caused a depreciation in precision and accuracy and longer computation times as it became more difficult for the algorithm to converge. Reducing the dimensions of the feature space with MLKR increased computation time but did not improve the mean Average Precision.

5.4. Neural Network with Triplet loss

Another approach for distance metric learning includes the use of Neural Networks. A fully connected Neural Network with hidden layers was created using Tensorflow that takes all 2048 features as input and returns a reduced number of output features N_{out} that can be used for the standard kNN retrieval experiments. For training, it was important to select a suitable loss function. After a thorough literature research, we were encouraged to make use of Triplet Loss as it was successfully employed for similar person re-identification tasks before [3]. A suitable Python implementation of the batch hard triplet loss strategy was found online [6] and adapted to the specifications of our problem. In this implementation, during each training iteration the Neural Network is fed with a batch of randomly selected PK samples, where P is the number of IDs and K is the number of samples per ID. For each sample in the batch, the hardest triplet is found - i.e. three indices (i, j, k) from the batch so that $\mathbf{x_j}$ is the point with the same label and the furthest distance from $\mathbf{x_i}$ (hardest positive) and $\mathbf{x_k}$ is the closest sample with a different label (hardest negative). The Triplet loss is then defined as $\epsilon = \max(||(\mathbf{x_i}-\mathbf{x_j})|| - ||(\mathbf{x_i}-\mathbf{x_k})|| + \text{margin}, 0)$. This implementation keeps the number of considered triplets relatively small, thus increasing computational speed, while making a useful selection whose proneness to constant outlier selection can be decreased by keeping PK relatively small. The considered triplets can be viewed as *moderate*, as they are the hardest for a small subset, which is optimal for training [3]. After averaging over all samples, the Triplet loss is used in combination with the Adam Optimiser - an experimentally proven extension to the classical SGD that computes individual learning rates for each network weight [1] - to adjust the weights of the Neural Network. The architecture can be optimised for a wide range of hyperparameters and architectural design choices. The best possible result (mAP = 0.397) on the test set was achieved with a fully connected Neural Network with one input layer (2048 neurons), two hidden layers (1024 and 513 neurons respectively) and an output layer of 256 neurons. For the first hidden layer ReLU was found to be a suitable activation functions, while the other layers remained linear. To improve generalisation during training dropout with a probability of 50% was performed on the first hidden layer over the 10 000 training iterations. The optimal learning rate was found to be 10^{-4} and for each batch four samples for 72 different IDs were chosen. Using no activation functions but an additional third hidden layer led to only slightly worse results (0.3754), while using more layers with activation functions or the sigmoid activation function led to strong depreciation in performance (mAP = 0.1144) pointing to overfitting. While these results are not better than the once achieved with Mahalanobis distance metrics, they come comparatively close and it can be expected that better results could be potentially achieved with bigger computational capacity allowing to validate more parameters.

6. Evaluation and summary of results

As can be seen from Table 1, the best performance was achieved with the optimised MLKR method over almost all metrics considered. While this distance metric performed only very slightly better than the original baseline method, it was able to reduce the output feature space to a dimension of 50 allowing a speed up in computation of the retrieval experiments. The worst performance was achieved with the original covariance based Mahalanobis method which is expected due to its unsupervised nature. Moreover, it is worth to mention that the performance measured in terms of rank@1, rank@5, rank@10 and mAP follow a very similar trend over the different methods considered showing that mAP is a well-designed metric and was rightfully chosen as main comparison metric. MLKR and LMNN show notable improvement in terms of kmeans over the baseline approach underlining their positive effects in drawing samples to their cluster centroids. Finally, it becomes clear that the given dataset was already very well trained making it very difficult to achieve absolute improvements to the baseline for the given dataset (see Appendix E). In conclusion, the authors of this report were able to investigate various distance metric methods and apply them on a real-life problem.

Evaluation Method	rank1	rank5	rank10	mAP	kmeans
Baseline	0.470	0.668	0.749	0.435	0.709
Original Maha.	0.431	0.638	0.711	0.390	0.696
LMNN	0.454	0.659	0.736	0.411	0.719
MLKR	0.469	0.671	0.749	0.436	0.719
Neural Net.	0.438	0.635	0.731	0.398	0.691

Table 1. The final results obtained by the models with optimal hyperparameters and prior feature reduction (see Appendix D)

References

[1] J. Brownlee. Gentle introduction to the adam optimization algorithm for deep learning. 2017.

[2] C. J. Carey and Y. Tang. metric-learn: Metric learning in python, 2018.

[3] A. Hermans, L. Beyer, and B. Leibe. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*, 2017.

[4] W. Li, R. Zhao, T. Xiao, and X. Wang. Deepreid: Deep filter pairing neural network for person re-identification, 2014.

[5] P. C. Mahalanobis. On the generalized distance in statistics. National Institute of Science of India, 1936.

[6] O. Moindrot. Triplet loss and online triplet mining in tensorflow, 2018.

[7] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[8] S. E. Robertson. The methodology of information retrieval experiment, 1981.

[9] G. Singh. Introductory guide to information retrieval using knn and kdtree, 2017.

[10] K. Q. Weinberger, J. Blitzer, and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. In Y. Weiss, B. Schölkopf, and J. C. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 1473–1480. MIT Press, 2006.

[11] K. Q. Weinberger and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(Feb):207–244, 2009.

[12] K. Q. Weinberger and G. Tesauro. Metric learning for kernel regression. In *Artificial Intelligence and Statistics*, pages 612–619, 2007.

A. Explanations of different retrieval experiment metrics

- Precision: The proportion of pictures that are taken of the same person as the query image in the nearest neighbours.
- Recall: The proportion of the pictures in the gallery set with the same label as the query image that were selected as nearest neighbours.
- Average precision (AP): After having calculated precision and recall values for a range of k values, precision values are mapped to recall values in the (0, 0.1, 1) range. The average precision is computed as the average of maximum precision at these 11 recall levels.
- Mean average Precision (mAP): It is the mean of the average precisions calculated for all query images.
- rank@N: The proportion of query picture where the N nearest neighbours include a picture of the same ID as the query.
- CMC Curve: a graph of the rank@N values versus N.

As the CMC curves (see Figures 1 and 2) look very similar for all distance metric methods considered they were not really helpful in drawing comparisons. As expected, the CMC curve is a monotonically increasing function between 0 and 1.

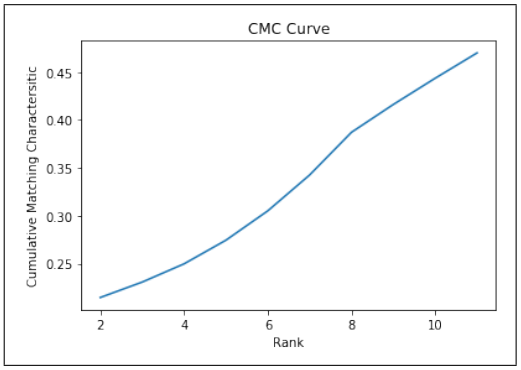

Figure 1. Baseline CMC Curve

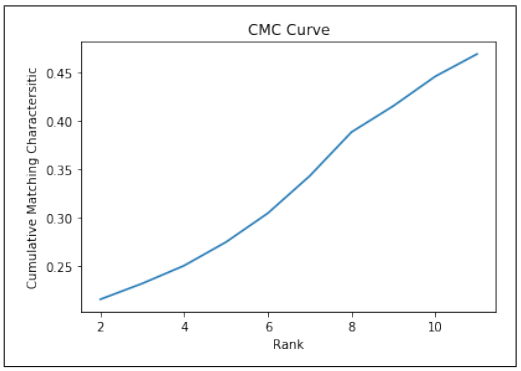

Figure 2. MLKR CMC Curve

B. PCA on training features

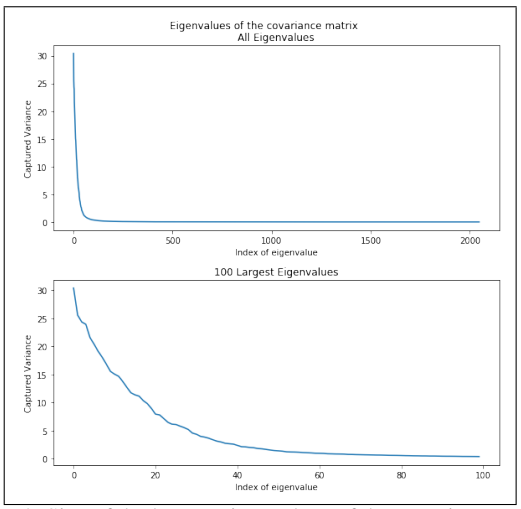

Figure 3. Size of the largest eigenvalues of the covariance matrix for PCA

C. Convex optimisation for LMNN

The optimisation of Equation (5.3.2) can be reformulated as follows [10]:
Minimise $\sum_{ij} \eta_{ij}(\mathbf{x_i}-\mathbf{x_j})^T A(\mathbf{x_i}-\mathbf{x_j}) + c\sum_{ijl} \eta_{ij}(1-y_{il})\xi_{ijl}$ **subject to:**

1. $(\mathbf{x_i}-\mathbf{x_l})^T A(\mathbf{x_i}-\mathbf{x_l})$-$(\mathbf{x_i}-\mathbf{x_j})^T A(\mathbf{x_i}-\mathbf{x_j}) \geq 1-\xi_{ijl}$
2. $\xi_{ijl} \geq 0$
3. $A \succeq 0$.

D. Hyperparameters for optimised methods

Evaluation Method	Hyperparameters and prior feature reduction
Baseline	Original feature space (similar with PCA)
Original	Input: PCA: 50 dimensions
Maha.	
LMNN	$k = 5$; Input: PCA: 500 dimensions
MLKR	Input: PCA: 500 dimensions, no further feat. reduction
Neural Net.	Triplet loss, 2 hidden layers, ReLU activation, dropout rate = 0.5, batch-size = 72

Table 2. Optimal hyperparameters and input features for the considered distance metric methods

E. Visual Analysis

From Figure 4, it can be seen that the retrieval experiment is heavily influenced by the clothing and hair colour of the pedestrians. The face cannot always be used for recognition purposes as it is often occluded. The person with the third query ID has a white T-Shirt and a bag that are easily recognisable from different angles. Indeed, the retrieval method performs very successfully (indicated by green boundary). However, the person with the 133th query ID wears a white shirt with a black coat. The wrongly retrieved images (red boundary) show pedestrians with very similar outfits which makes it quite difficult to distinguish between the different persons. In certain cases, it is expected that even a real person would have difficulties. For the given dataset, the person re-ID problem can be quite difficult due to possibly large inter-class similarities caused by similar clothing.

Figure 4. Baseline approach ranklist

F. Source Code

A zip of the executed Python Jupyter Notebook can be found via the link `https://imperialcollegelondon.box.com/s/ousbqh9kra6r8wm905yxe8fxa5lbyn0m`. All results should be already printed and viewable. All libraries used for the code can be installed via pip. For this project, Python 3.6.6 was used.