Torchreid: A Library for Deep Learning Person Re-Identification in Pytorch

Kaiyang Zhou **Tao Xiang**
University of Surrey, UK

`https://github.com/KaiyangZhou/deep-person-reid`

Abstract

Person re-identification (re-ID), which aims to re-identify people across different camera views, has been significantly advanced by deep learning in recent years, particularly with convolutional neural networks (CNNs). In this paper, we present `Torchreid`, a software library built on PyTorch that allows fast development and end-to-end training and evaluation of deep re-ID models. As a general-purpose framework for person re-ID research, Torchreid provides (1) unified data loaders that support 15 commonly used re-ID benchmark datasets covering both image and video domains, (2) streamlined pipelines for quick development and benchmarking of deep re-ID models, and (3) implementations of the latest re-ID CNN architectures along with their pre-trained models to facilitate reproducibility as well as future research. With a high-level modularity in its design, Torchreid offers a great flexibility to allow easy extension to new datasets, CNN models and loss functions.

1 Introduction

Driven by the growing demands for intelligent surveillance and forensic applications, person re-identification (re-ID) has become a topical research area in computer vision. This is evidenced by the increasing amount of research papers published in top-tier computer vision venues in recent years (see Figure 1). In particular, by digging into the title and abstract of the papers, we can observe a general trend that person re-ID research has moved from feature engineering (Liao et al., 2015; Matsukawa et al., 2016) and metric learning (Liao et al., 2015; Zhang et al., 2016), a two-stage pipeline, to end-to-end feature representation learning with deep neural networks (Li et al., 2014; Ahmed et al., 2015; Li et al., 2018; Chang et al., 2018; Zhou et al., 2019b; Chen et al., 2019a; Hou et al., 2019), particularly convolutional neural networks (CNNs). This can be attributed to the rapid advancement of deep learning technology, e.g., network architectures (He et al., 2016; Xie et al., 2017; Huang et al., 2017; Sandler et al., 2018; Zhang et al., 2018), optimisation/training techniques (Kingma and Ba, 2014; Reddi et al., 2018; Ioffe and Szegedy, 2015; Loshchilov and Hutter, 2017; Srivastava et al., 2014; Liu et al., 2019), as well as to the open-source deep learning frameworks, such as Caffe (Jia et al., 2014), PyTorch (Paszke et al., 2017), TensorFlow (Abadi et al., 2016) and MXNet (Chen et al., 2015), which enable researchers to quickly implement and test ideas and develop their own deep-learning projects.

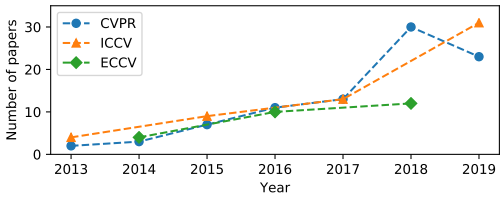

Figure 1: Number of papers on person re-ID that are published in top-tier computer vision conferences over 2013 - 2019.

Although the source code of some deep re-ID models has been released to the public, they typically differ in programming languages, backend frameworks, data-loading components, evaluation procedures, etc., which make the comparison between different approaches more difficult. This becomes more obvious when one wants to adapt the code of a published method to a new dataset or task for comparison, which may require a considerable amount of time spent on understanding and modifying the source code. This motivates us to design a generic framework that provides a standardised data-loading interface, basic training pipelines compatible with different re-ID models, and more importantly, is easy to extend.

Over the past decade, more than 30 person re-ID benchmark datasets have been introduced to the re-ID community[1], evolving from small datasets like VIPeR (Gray et al., 2007) and GRID (Loy et al., 2009) (with around thousands of images) to big datasets such as MSMT17 (Wei et al., 2018) (with over 100k images). However, even the largest re-ID dataset to date is still considered as being of moderate size when compared with contemporary large-scale datasets such as ImageNet (Deng et al., 2009). This is mainly due to the difficulty and expensive cost in collecting person images with pair-wise annotations across disjoint camera views. Therefore, it is common to combine different re-ID datasets for CNN model training (Xiao et al., 2016) or pre-training (typically followed by fine-tuning on small datasets (Li et al., 2017; Liu et al., 2017; Zhao et al., 2017; Zhou et al., 2019b)). From an engineering perspective, this requires the data loaders to accept an arbitrary number of training datasets and automatically adjust the identity and camera view labels to avoid conflict. Moreover, to allow evaluation of cross-dataset performance, the data loaders also need to be able to process test images from different datasets.

In this paper, we present `Torchreid`, which is a software library built on PyTorch to provide not only a unified interface for both image and video re-ID datasets, but also streamlined pipelines that allow fast development and end-to-end training and evaluation of deep re-ID models. Specifically, Torchreid supports 15 commonly used re-ID datasets, including 11 image datasets and 4 video datasets. Basically, users are allowed to select any number of (available) datasets of the same type (image or video) for model training and evaluation.

For CNN model learning, Torchreid currently implements two training pipelines, which are classification with `softmax`[2] loss and metric learning with `triplet`[3] loss, the two widely used (and most effective) objective functions in the literature. Users are allowed to choose either one of them or a weighted combination. Furthermore, Torchreid contains implementations of the latest state-of-the-art re-ID CNNs, together with their pre-trained models publicly available in Torchreid's model zoo[4]. Therefore, with Torchreid one can quickly get hands-on experience to train a strong baseline model or build on the current state of the arts for further research. More importantly, following the principle of code reusability and structural modularity, the training pipelines are carefully designed and structured such that a new pipeline can be efficiently constructed without much code re-writing.

Besides the efficient data loading, training and evaluation procedures, Torchreid also provides a visualisation toolkit to aid the understanding of re-ID model learning, e.g., visualisation of ranking results and activation maps, as well as a full documentation[5] including tutorials, how-to instructions, package references, etc. to help users quickly get familiar with the library.

The rest of the paper is organised as follows. Section 2 gives an overview of the Torchreid library. Section 3 introduces Torchreid's main modules. Section 4 concludes this paper with a discussion.

2 Overview of Torchreid

Torchreid is a library specifically developed for deep learning and person re-ID research, with the goal of providing an easy-to-use framework for benchmarking deep re-ID models and facilitating future research in re-ID. This library is written in Python, with some code based on Cython for

[1] See `https://github.com/NEU-Gou/awesome-reid-dataset` for a nice summary.

[2] Cross-entropy loss.

[3] Hard example mining triplet loss (Hermans et al., 2017).

[4] Torchreid's model zoo: `https://kaiyangzhou.github.io/deep-person-reid/MODEL_ZOO`.

[5] Torchreid's documentation: `https://kaiyangzhou.github.io/deep-person-reid/`.

```
torchreid/
  data/ # data loaders, data augmentation methods, data samplers
  engine/ # training and evaluation pipelines
  losses/ # loss functions
  metrics/ # distance metrics, evaluation metrics
  models # CNN architectures
  optim/ # optimiser and learning rate schedulers
  utils/ # useful tools (also suitable for other PyTorch projects)
```

Listing 1: Structure of Torchreid library.

optimisation and acceleration[6], and is built on top of PyTorch for automatic differentiation and fast tensor computation on GPUs.

The overall structure is shown in Listing 1. There are totally 6 modules among which we will discuss 3 main modules in more details in the next section, namely `data`, `engine` and `models`. The library is fully documented and is easy to install. It also hosts a model zoo with various pre-trained re-ID CNN models publicly available for download.

3 Main Modules

3.1 Data

Perhaps during the implementation of a computer vision project (or more generally a machine learning project), the most effort is not devoted to the implementation of a particular model or training algorithm, but to the construction of unified data loaders for preparing data. In Torchreid, we build each dataset on top of base classes (`ImageDataset` and `VideoDataset`), which provide basic functions for sampling, reading and pre-processing images. Therefore, for a new dataset users only need to define its training, query and gallery image sets, more specifically, the image paths, identity and camera view labels. To allow seamless combination of different re-ID datasets, the base classes are implemented with a customised `add` function such that different dataset *instances* can be directly summed up to obtain the combined dataset.

The training and test data loaders are wrapped in a high-level class called `DataManager`, which is responsible for the construction of sampling strategy, data augmentation methods and data loaders. An instance of `DataManager` will serve as the input to the training pipeline. We have `ImageDataManager` for image datasets and `VideoDataManager` for video datasets, both being the child classes of `DataManager`.

Currently, Torchreid supports the following re-ID datasets that are commonly used in the literature,

- **Image datasets**: Market1501 (Zheng et al., 2015), CUHK03 (Li et al., 2014), DukeMTMC-reID (Ristani et al., 2016; Zheng et al., 2017), MSMT17 (Wei et al., 2018), VIPeR (Gray et al., 2007), GRID (Loy et al., 2009), CUHK01 (Li et al., 2012), SenseReID (Zhao et al., 2017), QMUL-iLIDS (Zheng et al., 2009), PRID (Hirzer et al., 2011a), and CUHK02 (Li and Wang, 2013).
- **Video datasets**: MARS (Zheng et al., 2016), iLIDS-VID (Wang et al., 2014), PRID2011 (Hirzer et al., 2011b), and DukeMTMC-VideoReID (Ristani et al., 2016; Wu et al., 2018).

All datasets are implemented with their *de facto* evaluation protocols so that the evaluation results can be fairly compared with published papers.

Listing 2 (step 2) shows an example of how to construct a `ImageDataManager` to do model training and evaluation on Market1501. Note that the `sources` and `targets` attributes can take any number of datasets as input, as long as the datasets are available. For example, when `sources=['market1501',`

[6]Notably, with our Cython code the calculation of CMC ranks and mAP can be greatly accelerated, e.g., 3 mins shortened to 7 s on Market1501 (Zheng et al., 2015), 30 mins shortened to 3 mins on MSMT17 (Wei et al., 2018).

```
# Step 1: import the Torchreid library
import torchreid

# Step 2: construct data manager
datamanager = torchreid.data.ImageDataManager(
    root='reid-data',
    sources='market1501',
    targets='market1501',
    height=256,
    width=128,
    batch_size_train=32,
    batch_size_test=100,
    transforms=['random_flip', 'random_crop']
)

# Step 3: construct CNN model
model = torchreid.models.build_model(
    name='resnet50',
    num_classes=datamanager.num_train_pids,
    loss='softmax',
    pretrained=True
)
model = model.cuda()

# Step 4: initialise optimiser and learning rate scheduler
optimizer = torchreid.optim.build_optimizer(
    model,
    optim='adam',
    lr=0.0003
)

scheduler = torchreid.optim.build_lr_scheduler(
    optimizer,
    lr_scheduler='single_step',
    stepsize=20
)

# Step 5: construct engine
engine = torchreid.engine.ImageSoftmaxEngine(
    datamanager,
    model,
    optimizer=optimizer,
    scheduler=scheduler,
    label_smooth=True
)

# Step 6: run model training and test
engine.run(
    save_dir='log/resnet50',
    max_epoch=60,
    eval_freq=10,
    print_freq=10,
    test_only=False
)
```

Listing 2: Example of using the high-level APIs in Torchreid for model training and test.


Figure 2: A visualised ranking list using Torchreid. The black, green and red colours outline the query image, correct matches and false matches, respectively.

`'dukemtmcreid']`, the training data will be the combined training images from Market1501 and DukeMTMC-reID; when `targets=['market1501', 'dukemtmcreid']`, it will return the query and gallery images from Market1501 and DukeMTMC-reID separately.

3.2 Engine

The `engine` module, which aims to provide a streamlined pipeline for training and evaluation of deep re-ID models, is carefully designed to be as modular as possible for easy extension. Concretely, a generic base `Engine` is implemented for both image- and video-based re-ID to provide universal training loops and other reusable features, such as data parsing, model checkpointing and performance measurement. Therefore, new engines can subclass this `Engine` to reduce tedious code re-writing.

Based on the `Engine`, two learning paradigms are currently implemented for CNN model training, one is classification with `softmax` loss (`ImageSoftmaxEngine` & `VideoSoftmaxEngine`) and the other is metric learning with `triplet` loss (`ImageTripletEngine` & `VideoTripletEngine`). These two learning paradigms have been widely adopted in recent top-performing re-ID models (Hermans et al., 2017; Li et al., 2018; Chang et al., 2018; Zhou et al., 2019b; Chen et al., 2019a; Zhou et al., 2019a). An example of how to use this module is illustrated in steps 5 & 6 of Listing 2.

Besides the basic features necessary to construct a full train-evaluation pipeline, Torchreid also provides some advanced training tricks to improve the re-ID performance. For instance, to reduce overfitting the label smoothing regulariser (Szegedy et al., 2016) is implemented for the `softmax` pipeline; for better transfer learning the pipeline allows the pre-trained CNN layers to be frozen during early training (Geng et al., 2016) where the layers are specified by users.

Visualisation toolkit As qualitative result is often easier for us to understand how well a CNN model has learned, we implement two visualisation functions in Torchreid. The first function is `visrank`, which can visualise the ranking result of a re-ID CNN by saving for each query image the top-k similar gallery images (k is decided by users). Figure 2 shows an example of the visualised ranking list.

The second function is `visactmap`, which stands for visualising activation maps. Given an input image, the activation map can be used to analyse where the CNN focuses on to extract features (Zhou et al., 2019b). Specifically, an activation map is computed by taking the sum of absolute-valued feature maps (typically the highest-level feature maps) along the channel dimension, followed by a spatial $\ell 2$ normalisation (Zagoruyko and Komodakis, 2017). An example is shown in Figure 3, which is obtained by OSNet (Zhou et al., 2019b,a). Intuitively, the image regions with warmer colours have higher activation values, which contribute the most to the generation of final feature vectors. Whereas the regions with cold colours are likely to contain less important/reliable regions for re-ID.

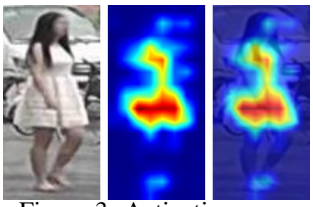
Figure 3: Activation map.

In addition, Torchreid is also integrated with PyTorch's built-in `SummaryWriter` for tensorboard[7] visualisation. For more features, please visit our github repository: `https://github.com/KaiyangZhou/deep-person-reid`.

3.3 Models

The `models` package provides a collection of implementations of state-of-the-art CNN architectures, which includes not only models that are specifically designed for re-ID, but also generic object

[7] `https://www.tensorflow.org/tensorboard/`.

recognition models that have been widely used as the re-ID CNN backbones. The currently available models are listed below,

- **ImageNet classification models**: ResNet (He et al., 2016), ResNeXt (Xie et al., 2017), SENet (Hu et al., 2018), DenseNet (Huang et al., 2017), Inception-ResNet-V2 (Szegedy et al., 2017), Inception-V4 (Szegedy et al., 2017), and Xception (Chollet, 2017).
- **Lightweight models**: NASNet (Zoph et al., 2018), MobileNetV2 (Sandler et al., 2018), ShuffleNet(V2) (Zhang et al., 2018; Ma et al., 2018), and SqueezeNet (Iandola et al., 2016).
- **Re-ID specific models**: MuDeep (Qian et al., 2017), ResNet-mid (Yu et al., 2017), HACNN (Li et al., 2018), PCB (Sun et al., 2018), MLFN (Chang et al., 2018), OSNet (Zhou et al., 2019b), and OSNet-AIN (Zhou et al., 2019a).

In addition to the CNN model code, we also release the corresponding model weights trained on the re-ID datasets (as well as the ImageNet pre-trained weights for some models) to further facilitate person re-ID research. We will keep updating this `models` package by incorporating new CNN models.

4 Discussion

Open-source frameworks are vital for pushing forward the progress of deep learning (DL) research. These include not only the backend engines for DL frameworks, such as PyTorch and TensorFlow, but also their extended libraries and toolkits that constitute the DL ecosystems, covering different research areas and providing convenient tools for fast research prototyping and development. In particular, recent years have witnessed an emergence of remarkable open-source frameworks/libraries developed for a wide range of research topics, e.g., Detectron (Girshick et al., 2018), MMDetection (Chen et al., 2019b) and SimpleDet (Chen et al., 2019c) for object detection, MMAction (Yue Zhao, 2019) for action recognition, AllenNLP (Gardner et al., 2018) and Transformers (Wolf et al., 2019) for natural language processing, Torchmeta (Deleu et al., 2019) for meta-learning, etc.

These open-source projects can greatly reduce the efforts for researchers and practitioners to reproduce state-of-the-art models and moreover, provide streamlined pipelines for easy development and extension. With a similar goal, Torchreid is specifically designed for DL and person re-ID research. Though there exist some excellent open-source projects for re-ID, such as `Open-ReID`[8] and `Person_reID_baseline_pytorch`[9], Torchreid is clearly different due to its unique features and versatility. In the future, we are committed to maintaining Torchreid and keeping it up-to-date by, for example, adding new datasets, CNN models and training algorithms.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., et al. (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*.

Ahmed, E., Jones, M., and Marks, T. K. (2015). An improved deep learning architecture for person re-identification. In *CVPR*.

Chang, X., Hospedales, T. M., and Xiang, T. (2018). Multi-level factorisation net for person re-identification. In *CVPR*.

Chen, B., Deng, W., and Hu, J. (2019a). Mixed high-order attention network for person re-identification. In *ICCV*.

Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Xu, J., Zhang, Z., Cheng, D., Zhu, C., Cheng, T., Zhao, Q., Li, B., Lu, X., Zhu, R., Wu, Y., Dai, J., Wang, J., Shi, J., Ouyang, W., Loy, C. C., and Lin, D. (2019b). MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*.

Chen, T., Li, M., Li, Y., Lin, M., Wang, N., Wang, M., Xiao, T., Xu, B., Zhang, C., and Zhang, Z. (2015). Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv preprint arXiv:1512.01274*.

[8] `https://github.com/Cysu/open-reid`.

[9] `https://github.com/layumi/Person_reID_baseline_pytorch`.

Chen, Y., , Han, C., Li, Y., Huang, Z., Jiang, Y., Wang, N., and Zhang, Z. (2019c). Simpledet: A simple and versatile distributed framework for object detection and instance recognition. *arXiv preprint arXiv:1903.05831*.

Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. In *CVPR*.

Deleu, T., Würfl, T., Samiei, M., Cohen, J. P., and Bengio, Y. (2019). Torchmeta: A Meta-Learning library for PyTorch. Available at: https://github.com/tristandeleu/pytorch-meta.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *CVPR*.

Gardner, M., Grus, J., Neumann, M., Tafjord, O., Dasigi, P., Liu, N., Peters, M., Schmitz, M., and Zettlemoyer, L. (2018). Allennlp: A deep semantic natural language processing platform. *arXiv preprint arXiv:1803.07640*.

Geng, M., Wang, Y., Xiang, T., and Tian, Y. (2016). Deep transfer learning for person re-identification. *arXiv preprint arXiv:1611.05244*.

Girshick, R., Radosavovic, I., Gkioxari, G., Dollár, P., and He, K. (2018). Detectron. `https://github.com/facebookresearch/detectron`.

Gray, D., Brennan, S., and Tao, H. (2007). Evaluating appearance models for recognition, reacquisition, and tracking. In *PETS*.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *CVPR*.

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

Hirzer, M., Beleznai, C., Roth, P. M., and Bischof, H. (2011a). Person Re-Identification by Descriptive and Discriminative Classification. In *SCIA*.

Hirzer, M., Beleznai, C., Roth, P. M., and Bischof, H. (2011b). Person re-identification by descriptive and discriminative classification. In *SCIA*.

Hou, R., Ma, B., Chang, H., Gu, X., Shan, S., and Chen, X. (2019). Interaction-and-aggregation network for person re-identification. In *CVPR*.

Hu, J., Shen, L., and Sun, G. (2018). Squeeze-and-excitation networks. In *CVPR*.

Huang, G., Liu, Z., Weinberger, K. Q., and van der Maaten, L. (2017). Densely connected convolutional networks. In *CVPR*.

Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., and Keutzer, K. (2016). Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size. *arXiv preprint arXiv:1602.07360*.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., and Darrell, T. (2014). Caffe: Convolutional architecture for fast feature embedding. In *ACM MM*.

Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Li, W. and Wang, X. (2013). Locally aligned feature transforms across views. In *CVPR*.

Li, W., Zhao, R., and Wang, X. (2012). Human reidentification with transferred metric learning. In *ACCV*.

Li, W., Zhao, R., Xiao, T., and Wang, X. (2014). Deepreid: Deep filter pairing neural network for person re-identification. In *CVPR*.

Li, W., Zhu, X., and Gong, S. (2017). Person re-identification by deep joint learning of multi-loss classification. In *IJCAI*.

Li, W., Zhu, X., and Gong, S. (2018). Harmonious attention network for person re-identification. In *CVPR*.

Liao, S., Hu, Y., Zhu, X., and Li, S. Z. (2015). Person re-identification by local maximal occurrence representation and metric learning. In *CVPR*.

Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., and Han, J. (2019). On the variance of the adaptive learning rate and beyond. *arXiv preprint arXiv:1908.03265*.

Liu, X., Zhao, H., Tian, M., Sheng, L., Shao, J., Yi, S., Yan, J., and Wang, X. (2017). Hydraplus-net: Attentive deep features for pedestrian analysis. In *ICCV*.

Loshchilov, I. and Hutter, F. (2017). Sgdr: Stochastic gradient descent with warm restarts. In *ICLR*.

Loy, C. C., Xiang, T., and Gong, S. (2009). Multi-camera activity correlation analysis. In *CVPR*.

Ma, N., Zhang, X., Zheng, H.-T., and Sun, J. (2018). Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *ECCV*.

Matsukawa, T., Okabe, T., Suzuki, E., and Sato, Y. (2016). Hierarchical gaussian descriptor for person re-identification. In *CVPR*.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in pytorch. In *NIPS-W*.

Qian, X., Fu, Y., Jiang, Y.-G., Xiang, T., and Xue, X. (2017). Multi-scale deep learning architectures for person re-identification. In *ICCV*.

Reddi, S. J., Kale, S., and Kumar, S. (2018). On the convergence of adam and beyond. In *International Conference on Learning Representations*.

Ristani, E., Solera, F., Zou, R., Cucchiara, R., and Tomasi, C. (2016). Performance measures and a data set for multi-target, multi-camera tracking. In *ECCVW*.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *JMLR*.

Sun, Y., Zheng, L., Yang, Y., Tian, Q., and Wang, S. (2018). Beyond part models: Person retrieval with refined part pooling (and a strong convolutional baseline). In *ECCV*.

Szegedy, C., Ioffe, S., Vanhoucke, V., and Alemi, A. A. (2017). Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *CVPR*.

Wang, T., Gong, S., Zhu, X., and Wang, S. (2014). Person re-identification by video ranking. In *ECCV*.

Wei, L., Zhang, S., Gao, W., and Tian, Q. (2018). Person transfer gan to bridge domain gap for person re-identification. In *CVPR*.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., and Brew, J. (2019). Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*.

Wu, Y., Lin, Y., Dong, X., Yan, Y., Ouyang, W., and Yang, Y. (2018). Exploit the unknown gradually: One-shot video-based person re-identification by stepwise learning. In *CVPR*.

Xiao, T., Li, H., Ouyang, W., and Wang, X. (2016). Learning deep feature representations with domain guided dropout for person re-identification. In *CVPR*.

Xie, S., Girshick, R., Dollár, P., Tu, Z., and He, K. (2017). Aggregated residual transformations for deep neural networks. In *CVPR*.

Yu, Q., Chang, X., Song, Y.-Z., Xiang, T., and Hospedales, T. M. (2017). The devil is in the middle: Exploiting mid-level representations for cross-domain instance matching. *arXiv preprint arXiv:1711.08106*.

Yue Zhao, Yuanjun Xiong, D. L. (2019). Mmaction. `https://github.com/open-mmlab/mmaction`.

Zagoruyko, S. and Komodakis, N. (2017). Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR*.

Zhang, L., Xiang, T., and Gong, S. (2016). Learning a discriminative null space for person re-identification. In *CVPR*.

Zhang, X., Zhou, X., Lin, M., and Sun, J. (2018). Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *CVPR*.

Zhao, H., Tian, M., Sun, S., Shao, J., Yan, J., Yi, S., Wang, X., and Tang, X. (2017). Spindle net: Person re-identification with human body region guided feature decomposition and fusion. In *CVPR*.

Zheng, L., Bie, Z., Sun, Y., Wang, J., Su, C., Wang, S., and Tian, Q. (2016). Mars: A video benchmark for large-scale person re-identification. In *ECCV*.

Zheng, L., Shen, L., Tian, L., Wang, S., Wang, J., and Tian, Q. (2015). Scalable person re-identification: A benchmark. In *ICCV*.

Zheng, W.-S., Gong, S., and Xiang, T. (2009). Associating groups of people. In *BMVC*.

Zheng, Z., Zheng, L., and Yang, Y. (2017). Unlabeled samples generated by gan improve the person re-identification baseline in vitro. In *ICCV*.

Zhou, K., Yang, Y., Cavallaro, A., and Xiang, T. (2019a). Learning generalisable omni-scale representations for person re-identification. *arXiv preprint arXiv:1910.06827*.

Zhou, K., Yang, Y., Cavallaro, A., and Xiang, T. (2019b). Omni-scale feature learning for person re-identification. In *ICCV*.

Zoph, B., Vasudevan, V., Shlens, J., and Le, Q. V. (2018). Learning transferable architectures for scalable image recognition. In *CVPR*.