# Lossy Intermediate Deep Learning Feature Compression and Evaluation

Zhuo Chen
Nanyang Technological University
Singapore

Kui Fan
Peking University
Shenzhen, China

Shiqi Wang
City University of Hong Kong
Hong Kong, China

Ling-Yu Duan*
lingyu@pku.edu.cn
Peking University
Beijing, China

Weisi Lin*
WSLin@ntu.edu.sg
Nanyang Technological University
Singapore

Alex C. Kot
Nanyang Technological University
Singapore

## ABSTRACT

With the unprecedented success of deep learning in computer vision tasks, many cloud-based visual analysis applications are powered by deep learning models. However, the deep learning models are also characterized with high computational complexity and are task-specific, which may hinder the large-scale implementation of the conventional data communication paradigms. To enable a better balance among bandwidth usage, computational load and the generalization capability for cloud-end servers, we propose to compress and transmit intermediate deep learning features instead of visual signals and ultimately utilized features. The proposed strategy also provides a promising way for the standardization of deep feature coding. As the first attempt to this problem, we present a lossy compression framework and evaluation metrics for intermediate deep feature compression. Comprehensive experimental results show the effectiveness of our proposed methods and the feasibility of the proposed data transmission strategy. It is worth mentioning that the proposed compression framework and evaluation metrics have been adopted into the ongoing AVS (Audio Video Coding Standard Workgroup) - Visual Feature Coding Standard.

## CCS CONCEPTS

• **Information systems** → **Multimedia streaming**; • **General and reference** → Metrics; • **Computing methodologies** → Distributed artificial intelligence; Computer vision.

## KEYWORDS

deep learning; data transmission; feature coding

**ACM Reference Format:**
Zhuo Chen, Kui Fan, Shiqi Wang, Ling-Yu Duan, Weisi Lin, and Alex C. Kot. 2019. Lossy Intermediate Deep Learning Feature Compression and Evaluation. In *Proceedings of the 27th ACM International Conference on Multimedia (MM '19), October 21–25, 2019, Nice, France.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3343031.3350849

*Ling-Yu Duan and Weisi Lin are the corresponding authors.


*MM '19, October 21–25, 2019, Nice, France*

ACM ISBN 978-1-4503-6889-6/19/10…$15.00
https://doi.org/10.1145/3343031.3350849

## 1 INTRODUCTION

With the advances of network infrastructure, recent years has witnessed the explosive growth of cloud-based visual analysis applications, like surveillance analysis, smart city, visual-based positioning, autopilot, etc. In cloud-based visual analyses, visual signals are acquired by front-end devices and the analyses are finished in the cloud-end servers. With deep learning models showing incomparable performance in computer vision since 2012 [19], cloud-based visual analyses are increasingly relying on deep neural networks (DNNs), such as object detection [14, 27], vehicle [22] and person re-identification (ReID) [40], vehicle license plate recognition [26], face recognition [34, 35], pedestrian detection [25], landmark retrieval [38], autopilot [1], etc.

As to data communication between the front end and server end, the most conventional paradigm is known as "compress-then-analyse", shown as Figure 1(a). The visual signal is captured and compressed in the front-end devices, then the coding bitstream is conveyed to the cloud-end servers. Subsequently, visual analysis tasks will be performed in cloud-end servers according to the decoded visual signal. As the fundamental infrastructure of the paradigm, image/video compression has been well developed and matured. As the current generation video coding standard, High Efficiency Video Coding (HEVC) [33] achieves a half bit-rate reductions at equal perceptual visual quality level comparing to the last generation H.264/MPEG-4 Advanced Video Coding (AVC) [39]. The next generation video coding standardization, Versatile Video Coding (VVC) [2], is ongoing, which is expected to be completed before 2020 and has already achieved superior performance to HEVC.

Although supported by well developed standards and infrastructures, such paradigm is questionable when the system gets scaled up. In application scenarios like Internet-of-Things (IoT) and video surveillance, thousands-of-thousands front-end cameras can simultaneously produce large amount of visual signals. The bandwidth will be a bottleneck as signal level compression suffers from high transmission burden. Furthermore, the feature extraction of visual signals is computational intensive, especially with deep neural networks, which makes it unaffordable to simultaneously analyse large scale visual data in cloud-end servers.

An alternative way is "analyze-then-compress" [28], shown as Figure 1(b). With this strategy, both data acquisition and feature

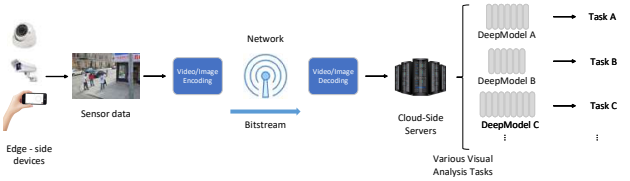
extraction occur in front-end devices, and only the ultimately utilized features (denoted as "ultimate features" in the rest of paper) are compressed and transmitted to the cloud. It provides a feasible solution for large scale cloud-based visual analysis systems, as the ultimate feature is compact and able to be utilized for analyses straightforwardly at the cloud end. Moreover, features, especially high-level features, are commonly extracted to reflect abstract semantic meaning, which largely eliminates visible information from the input signals. As such, the risk of privacy disclosure can be easily controlled by conveying features instead of signal-level data communication. Such paradigm is also supported by several feature coding standards, which are mainly for handcrafted ultimate features. In the context of image retrieval applications, Compact Descriptors for Visual Search (CDVS) [9] was published by Moving Picture Experts Group (MPEG) in 2015. Built upon CDVS, Compact Descriptors standardization for Video Analysis (CDVA) [9] was proposed by MPEG to deal with video retrieval applications. It is worth noting that working draft of CDVA also incorporated with deep features to accelerate retrieval performance [21].

Although the data transmission strategy of conveying ultimate features enjoys a good deal of favorable characteristics, one obstacle that may hinder the practical implementation of ultimate feature communication is that ultimate features are usually task-specific which makes the transmitted features hard to be applied to various analysis tasks. This may also impede the further standardization for deep feature coding, as the standardized deep features are expected to be well generalized to ensure the interoperability in various application scenarios.

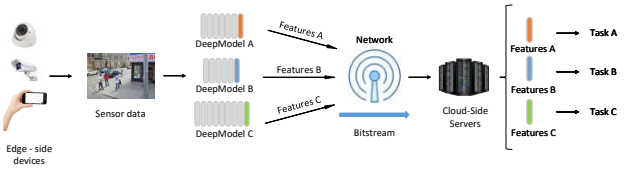
In view of the pros and cons of aforementioned two paradigms (summarized in Table 1), this paper proposed a new strategy of transmitting intermediate layer features of deep learning models (noted as "intermediate deep features" in the rest of the paper) instead of visual signals and ultimate features, which can achieve a balance among the computing load, communication cost and the generalization ability. As the infrastructure of this new strategy, intermediate deep feature compression has not been well explored in literature. As far as we know, there are only two works [6, 7] investigating intermediate deep feature compression on two specific types of features in the context of collaborative intelligence and image object detection. Problems like "how to efficiently compress intermediate deep features from different layers of different deep models with a unified compression framework" and "how to evaluate the compression methods" are still left open for exploration.

As a first attempt to the problem, in this paper we

- present and analyze the new data communication strategy of transmitting intermediate deep features for cloud-based visual analysis applications, which enables a good balance among the transmission load, computing load and the generalization ability for cloud servers; and
- propose a video codec based lossy compression framework for intermediate deep feature coding, which can provide good performance and make full use of the video coding infrastructures when upgrading the communication system; and
- introduce new metrics for fidelity evaluation of intermediate deep feature compression methods, and report comprehensive experimental results.




(a) Visual signal transmission. By transmitting the visual signal, a series of visual analysis tasks can be performed in the cloud. As such, the computing load including feature extraction and analysis is imposed on the cloud side.




(b) Ultimate feature transmission. Computing load can be distributed to front-end devices. Only specific types of analysis can be performed at the server-end, depending on the deep models used at the front-end.

**Figure 1: Two commonly used data transmission paradigms for cloud-based visual analysis.**

It is worth noting that the proposed lossy compression framework and the evaluation metrics has been adopted into AVS - Visual Feature Coding Standard [42].

The rest of the paper is organized as follows. Section 2 gives a detailed description on the proposed data transmission approach and envision the future standardization for intermediate deep feature coding; Section 3 presents our proposed lossy intermediate deep feature compression framework and the evaluation metrics; Section 4 reports exhaustive experimental results on the proposed methods and metrics; Section 5 concludes this paper.

## 2 TOWARD TRANSMISSION AND COMPRESSION OF INTERMEDIATE DEEP FEATURES

### 2.1 Intermediate deep feature transmission

In the context of cloud-based visual analysis, visual signal acquisition and analysis are processed in distributed devices. Sensor data (i.e., images and videos) are captured at front-end, like surveillance cameras and smart phones, while the analyses are completed in the cloud-end servers. Conventionally, the data communication between the front and cloud ends can be with either visual signals or ultimate features, as shown in Figure 1.

As discussed in Section 1, in manner of transmitting visual signal (shown as Figure 1(a)), all types of visual analyses, including manual monitoring, can be applied at the cloud end, since the image/video data are available. However, due to the visual signal degradation resulting from lossy image/video compression, the performance drop of the analysis tasks is nonnegligible, especially when the compression ratio is high [8, 20]. Moreover, it is doubtable that whether such signal level communication can efficiently handle the visual big data, as all the computing load for visual analyses allocates on the cloud-end servers. In term of tranmitting ultimate feature (shown as Figure 1(b)), the computing load on cloud side

**Table 1: Comparison of three transmission strategies**

| | Transmit Video Signal | Transmit Intermediate Feature | Transmit Ultimate Feature |
|---|---|---|---|
| Load allocation | All the computing load is on cloud end | The computing load can be largely shifted to the front end | |
| Usability | The transmitted signal can be applied to various tasks | | The transmitted feature can be only applied to few specific tasks |
| Privacy | Visual signal can easily expose privacy of users | It is difficult to unscramble features, especially when without corresponding models | |
| Research Status | Well explored and standardized (AVC, HEVC, VVC, etc.) | Research gap | Well explored and standardized (MPEG-CDVA, CDVS, etc.) |

can be largely shifted to the front-end devices, which makes cloud-based visual analysis feasible in the context of big data. However, as deep learning models are trained with a data-driven manner, the top-layer features are usually task-specific and hard to generalize to different visual analysis tasks. To enable multiple types of analysis in the cloud side, various deep learning models should be deployed in the front-end device, which makes the whole system bloated and complicated. In other words, the availability of visual analysis applications in cloud-end servers is constrained by the deep learning models implemented in the front-end devices.

As shown in Figure 2, to balance the computing load between the front and cloud ends without limiting the analysis capability in the cloud side, we propose to transmit the intermediate deep features instead of visual signals and ultimate features. Deep neural networks are with hierarchical structures, which can be considered as a combination of cascade feature extractors rather than a single straightforward feature extractor. The features from upper layers are more abstract and task-specific, whilethe features from lower layers can be applied to a broader range of analysis tasks. As such, the cloud-end servers can request proper features from front end according to the analysis tasks. In this manner, a generic deep model whose features can be applied to different visual analysis tasks is preferred to be deployed in the front end. Lightweighted task-specific neural networks, which takes the transmitted intermediate features as input, will be implemented on the cloud side to perform various analysis tasks.

At present, some deep learning models like VGGNet and ResNet are widely adopted as the backbone networks in many computer vision tasks [3, 12–15, 21, 24, 29, 37, 41]. Task-specific networks are built on top of particular intermediate features of the backbone networks. Such backbone networks can be regarded as generic to deployed in the front end. Table 2 summarizes the usability of the intermediate deep features. From the table, most of task-specific networks prefer to take high layer features ($conv4$ or higher) as their input. Since the computing load are mainly laid on low layers in neural networks, it can help saving great computing cost for the server-end with our proposed strategy. Thus, our proposed strategy can help minimizing the computing load in the cloud end while maximizing the availability of various analysis applications. Furthermore, it is anticipated that the deep neural networks will be developed to more and more generic in the future. At that stage, the proposed strategy will have more advantages over the conventional ones.

## 2.2 Intermediate deep feature compression

As discussed previosly, conveying intermediate deep features instead of visual signals and ultimate features is advantageous at reducing the computing load at the cloud end while maintaining the availability of various visual analysis applications. Nevertheless, the transmission load for intermediate deep features is nonnegligible, which make it indispensable to develop compression methods for intermediate deep features.

**Table 2: The computing cost of nueral networks usually lays on lower layers, while most of visual applications prefer to utilize upper layer features. This provides evidences that transmitting intermediate features can help shift the majority of computing load while maintaining the data usability.**

| | FLOPs | | Usage |
|---|---|---|---|
| | VGGNet | ResNet | |
| $conv1$<br>$pool1$ | 1.94$G$(12.5%) | 0.12$G$(3.1%) | / |
| $conv2$<br>$pool2$ | 2.77$G$(30.5%) | 0.67$G$(20.4%)<br>/ | / |
| $conv3$<br>$pool3$ | 4.62$G$(60.3%) | 0.95$G$(45.0%)<br>/ | / |
| $conv4$ | 4.62$G$(90.2%) | 1.39$G$(81.0%) | captioning [15], QA [12] |
| $pool4$ | | / | tracking [37] |
| $conv5$ | 1.39$G$(99.2%) | 0.73$G$(99.9%) | captioning [41], QA [24] |
| $pool5$ | | | tracking [37], detection [13, 29], retrieval [21], QA [24] |
| $fc$ | 0.12$G$(100%) | 2.05$M$(100%) | detection [14], retrieval [3] |

By investigating in successful neural network architectures (backbone architectures), like AlexNet [19], VGGNet [32], ResNet [17] and DenseNet [18], it can be found that such network architectures share similar block-wise structures and feature shapes [5]. In convolutional neural networks (CNNs), intermediate deep features are mainly in forms of feature maps which are the combinations of stacked 2-D matrices. The height and width of the feature maps gradually get reduced along with the inference process. One or several layers can be composed as a block which halves the height and width of the feature maps. So, with the same input size, certain blocks of different network architectures shall provide the feature maps with identical height and width. Furthermore, the numerical distributions of intermediate deep features also share similar properties, as most CNN architectures use ReLU as the non-linear transformation function which clips the features into same numerical range. Such observations provide possibility that intermediate deep features of different network architectures can be compressed with an unified compression method.

## 2.3 Standardization of Intermediate Deep Feature Compression

As the fundamental infrastructures of the paradigms of transmitting visual signals and ultimate features, quite a few standards for

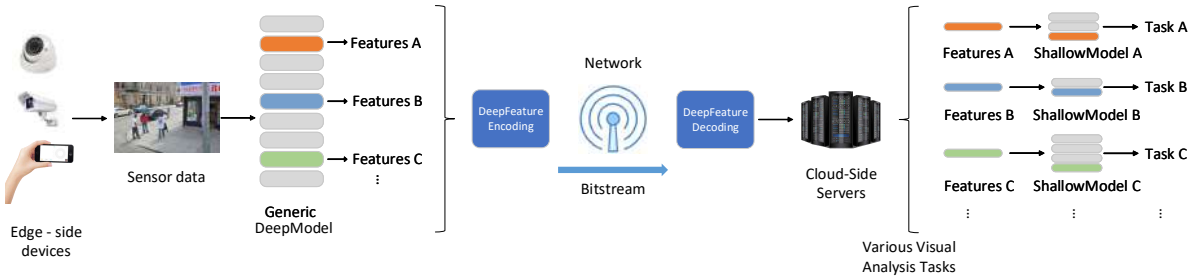



**Figure 2: Diagram of the proposed approach. The intermediate deep features of a generic deep model can be applied to a broad range of tasks. The features of specific layers will be transmitted based on the analysis requirements on the cloud side. On top of these transmitted features, shallow task-specific models will be applied for visual analysis.**

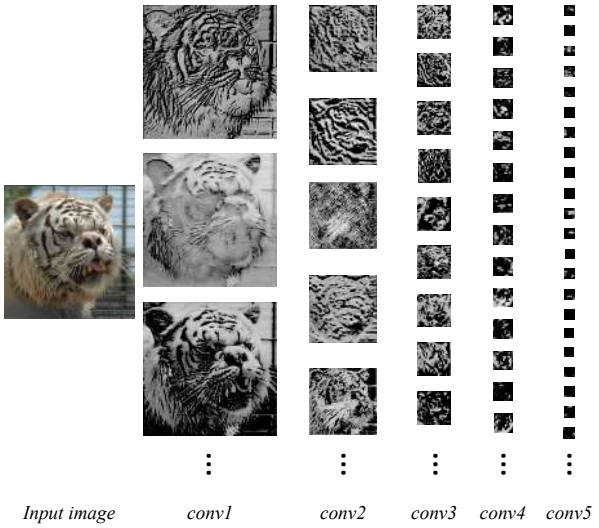



**Figure 3: Visualized feature maps of VGGNet.**

image/video coding and handcrafted feature coding have been published to ensure compatibility and interoperability, as mentioned in Section 1. It is also expected that intermediate deep feature coding can be standardized to facilitate the data communication of intermediate deep feature in cloud-based visual analysis applications.

Typically, feature coding standards, like CDVS [9] and CDVA [10], should specify both feature extraction and compression processes to fully ensure the interoperability, as features from different extractors may be with different shape, distribution and numerical type. With such standardization strategy, feature extractors are carefully designed and specified, which ensures the interoperability but sacrifices the compatibility for different feature extractors and the generality for different tasks. For intermediate deep feature coding, as discussed in Section 2.2, features from different deep learning models (feature extractors) share similar shapes and distributions, which make it possible to obtain the interoperability by only specifying the compression process. Since the choice of deep learning models are left open, the compatibility and generality of the standard can also be ensured together with the interoperability. Moreover, such standardization strategy is also good for keeping the standard with long-lasting vitality, as any new deep neural network with better performance in the future can be seamlessly adopted for system customization.

## 3 COMPRESSION AND EVALUATION METHODS

As presented in [5], the lossless compression methods can hardly achieve efficient compression with high compression ratio, which is not desirable for practical applications. In this paper, we study lossy compression for intermediate deep features.

### 3.1 Video codec based lossy compression

In CNNs, the intermediate features are mainly in the form of feature maps which are the combinations of stacked 2-D arrays with spatial correlations among the elements, as shown in Figure 3. Intuitively, a single channel 2-D feature map can be consider as a frame, while an intermediate deep feature can be considered as one video sequence. As such, existing video codecs can be applied to compress deep features in a lossy manner. In this paper, we propose a video codec based compression framework for intermediate deep feature coding.

By integrating video codecs into the proposed compression framework, matured video coding techniques can be borrowed to intermediate feature coding seamlessly. Furthermore, as video encoding/decoding modules (chips, IP cores, etc.) have already been widely deployed in many cloud based systems, it is economically and technically friendly to upgrade the visual devices and systems to support intermediate deep feature conveyance and analysis with our proposed framework.

Figure 4 describes the flow of the compression method. In the encoding phase, pre-quantization is first applied, as the numerical type of deep features are commonly not compatible with the input of video codecs. For instance, the vanilla VGGNets and ResNets features are in $float32$, while video codecs, such as HEVC, are designed for integer input with 8 or higher bit depth. Different quantizers can be applied based on the distribution analyses of intermediate features.

After quantization, the $N$ feature samples $\mathbb{R}^{N\times H\times W\times C}$ will be repacked into video-sequence-like format $\mathbb{R}^{H'\times W'\times NC}$ to fit the video codec input, where $H$ and $W$ are the height and width of the feature map, $C$ is the channel number (i.e. the number of feature maps) of the feature sample. As the input frame size of the video codec is usually non-arbitrary, like the input size of HEVC can only be integral multiple of 8, the original feature map size $H\times W$ should be extended to $H'\times W'$ by padding methods. Particularly, $H' = \lceil H/8\rceil \times 8$ and $W' = \lceil W/8\rceil \times 8$, where $\lceil\cdot\rceil$ is the ceiling operation. The order of the frames can be further reorganized during the

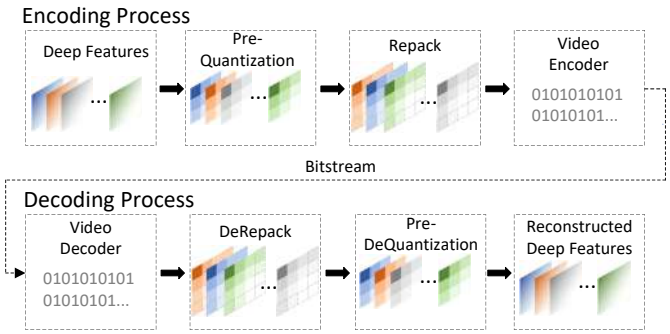



**Figure 4: Flowchart of lossy compression for intermediate deep feature maps.**

repack phase, as which may affect the compression performance if inter-frame correlations are considered. The repacked feature maps are then considered as 4:0:0 video sequences to feed into the video encoder.

In the decoding phase, the bitstream is first decoded by video decoder. Then, depack operation converts the reconstructed video-sequence-like data to the original feature size. Subsequently, the integer feature tensors are dequantized to float type. The reconstructed deep feature maps can then be passed to task-specific models to perform visual analyses.

## 3.2 Evaluation metrics

Similar to video coding, the evaluation of intermediate deep feature coding should take both compression performance and information loss into consideration. In this paper, compression rate is employed to evaluate the compression performance, which is defined as

$$Compression\ rate = \frac{data\ volume\ after\ compression}{data\ volume\ before\ compression}. \quad (1)$$

To evaluate information loss, the comparison of output results of the tasks performed after the feature transmission should be considered. It is because signal-level comparison (e.g., SNR, PSNR) for features is bootless, as deep features are with high level semantic information. It is also not proper to utilize task performance metrics (e.g., accuracy for image classification task, mAP for image retrieval task) to evaluate the performance of feature codecs. The reason is threefold. Firstly, the variation of a task performance metric may not reflect the fidelity level of the features before/after compression. Concretely, in terms of the direction of change, information loss of the features before/after compression can result in either positive or negative change to a task performance metric (e.g., classification accuracy varies from 0.80 to 0.75 or 0.85); in terms of the amount of change, same change amount of a task performance metric may refer to different information loss levels. The task performance metric may not be linearly proportional to inforation loss. Secondly, it is not well normalized to use task performance metrics to evaluate information loss. On the one hand, task performance metrics are with different value ranges (e.g., image classification accuracy is with the range of 0 to 1, while image captioning CIDEr [36] can reach more than 1); on the other hand, the task performance value on pristine features (i.e., the reference value) may vary depending on the test dataset, which makes it hard to compare information loss with task performance metrics. Thirdly, using the task performance metric to evaluate the information loss, paired values (before/after compression) should be involved which is not elegant.

Therefore, we propose new metrics to evaluate information loss of features on different tasks. In this paper, we choose three popular computer vision tasks in surveillance applications: image classification, image retrieval and image object detection, respectively. For image classification, we calculate the fidelity by comparing the pristine classification DNN outputs (i.e., the onehot classification results) with the outputs inferred from the reconstructed intermediate deep features, as below

$$F_{classification} = 1 - \frac{1}{2N}\sum_{i}^{N}\frac{Hamming(Y_p^i, Y_r^i)}{length(Y_p^i)} \quad (2)$$

where $Y_p^i$ is the pristine onehot output of the tested neural network inferred from $i$-th test image sample, $Y_r^i$ is the onehot output inferred from the corresponding reconstructed intermediate feature, $length(\cdot)$ returns the dimension of input, $N$ denotes the total number of tested samples.

For retrieval task, given a query, a ranked sequence of documents will be returned by the system. In task performance metrics like mean average precision (mAP), the order of the ranked sequence is taken into consideration to calculate the average precision (AP). Here, we propose to calculate the fidelity by comparing the pristine output document sequence with the one inferred from the reconstructed intermediate deep features:

$$F_{retrieval} = \frac{1}{N}\sum_{i}^{N} bubble\_index(S_p^i, S_r^i) \quad (3)$$

where $S_p^i$ and $S_r^i$ are the ranked sequence of documents returned by the retrieval system with pristine features and reconstructed respectively for the $i$-th query, $N$ denotes the total number of tested queries, $bubble\_index(\cdot,\cdot)$ is proposed to measure the similarity between two ranked sequences by counting the number of swap operations during sorting the reconstructed sequence into the pristine with bubble sort method. We name the similarity measurement after "bubble sort" method as "bubble index". The workflow of bubble index is described in Algorithm 1. It is worth noting that a naive implementation of bubble index is computational heavy ($O(n^2)$) especially when the length of input sequence is large. The computing complexity can be significantly reduced (less than $O(n\log(n))$) by applying dichotomy in the for-loop. The code implementation can be found at [1].

As to object detection task, the detection model predicts both location and category of detected objects. We use Intersection over Union (IoU) to measure the fidelity on the location of the predictions, and a relative change rate to monitor the predicted classification confidences. Moreover, considering that predictions with different confidence level contribute differently to the task performance, we weighted each prediction with the confidence inferred by the pristine feature. Overall, the fidelity in object detection task is

[1] http://chenzhuo.info/repos/acmmm19.html

**Algorithm 1** Calculate similarity between two ranked sequences of documents

**Input:** $S_p$: ranked sequence of documents returned by the retrieval system with pristine features; $S_r$: the sequence inferred from the reconstructed intermediate deep features.
**Output:** Similarity score

```
procedure BUBBLE_INDEX(S_p,S_r)
    for element in S_r do                          ▷ generate list to be sort
        ordered_list.append(S_p.index(element))
    cnt_swap, cnt_total = 0
    for i in range(len(ordered_list)-1) do         ▷ Bubble sort
        for j in range(len(ordered_list)-i-1) do
            cnt_total += 1
            if ordered_list[j] > ordered_list[j + 1] then
                cnt_swap += 1
                swap ordered_list[j] and ordered_list[j + 1]
    Similarity score = cnt_swap/cnt_total
```

calculated as below:

$$F_{detection} = \frac{1}{N}\sum_{i}^{N}\frac{\sum_{j}^{M} UoI(B_p^{ij}, B_r^{ij}) \cdot max(0, 1 - \frac{C_p^{ij} - C_r^{ij}}{C_p^{ij}}) \cdot C_p^{ij}}{\sum_{j}^{M} C_p^{ij}} \tag{4}$$

where $B$ is the predicted bounding box and $C$ is the confidence value of the predicted category, $N$ is the number of tested images and $M$ is the number of predicted objects of $i$-th image. The implement code can be found at [1].

## 4 EXPERIMENTAL RESULTS

To provide evidences for the feasibility of the strategy of transmitting intermediate deep feature and the effectiveness of the proposed lossy compression framework, we present comprehensive experiments of intermediate deep feature compression on three widely-used visual surveillance tasks with two commonly-used backbone neural networks.

### 4.1 Experiment Setup

*4.1.1 Evaluation tasks and datasets.* As discussed in Section 2, one advantage of our proposed data transmission strategy relies on that intermediate deep features are with good generic ability which can be applied to a broad range of tasks. As such, we compress the intermediate features from unified backbone networks, then evaluate the information loss on three notable tasks in visual surveillance, which are image classification, image retrieval and image object detection, respectively.

***Image classification:*** As a fundamental task in computer vision, image classification has been widely adopted in training and evaluating deep learning architectures. Many generic networks trained on image classification (e.g., VGGNet, ResNet) are employed as feature extractors or backbone networks in other computer vision tasks. Following [5], in this paper, we evaluate information loss in feature compression on image classification task with a subset of the validation set of the ImageNet 2012 dataset [30]. To economize the compression time while maintaining the variety of test image categories, we randomly choose one image from each of the 1,000 classes.

***Image retrieval:*** Content-based image retrieval is another key problem in computer vision. Among image retrieval problems, vehicle retrieval, as an unique application, has been drawing more and more attention due to the explosively growing requirement on surveillance security field. In this paper, we adopt the "Small" test split of PKU VehicleID dataset [22] to perform feature compression evaluation on image retrieval task, which contains 800 query images and 5,693 reference images. In the experiments, only the features extracted from query images are to be compressed. Features extracted from reference images serve as reference during fidelity evaluations.

***Image object detection:*** Image object detection task predicts both object location and category in the mean time, which contains both regression and classification. It is a fundamental task for surveillance analyses. We test our compression algorithm on image object detection with the test set of Pascal Visual Object Classes (VOC) 2007 dataset [11], which contains 4,952 images and 12,032 objects.

*4.1.2 Deep learning architectures and features.* In the experiments, we extract intermediate deep features with VGGNets and ResNets, which are the common choices for image feature extraction in many computer vision applications as their features can be regarded as generic.

***VGGNet:*** Simonyan and Zisserman developped VGGNet [32] at the ILSVRC 2014. VGGNet-16 outstands from the six variants of VGGNet for its good balance among performance and computing complexity. VGG-16 is very appealing thanks its neat architecture consisting of 16 convolutional layers which only performs $3 \times 3$ convolution and 2×2 pooling all the way through. Currently it is the most preferred choice to extract features from images in computer vision community. In this paper, we extract *conv*1 to *pool*5 features from VGGNet-16 architecture to compress and evaluate in image classification; *pool*3 and *pool*4 are not included in image retrieval task due to [23] perform feature downsampling in *conv*1 and *conv*2 by setting convolution stride instead of *pool*3 and *pool*4; *pool*5 is not included in detection task due to the region proposal network (RPN) of faster RCNN is built on top of *conv*5 feature of VGGNet. The implementation of image classification follows [31], image retrieval follows [23] and image object detection follows [4].

***ResNet:*** At the ILSVRC 2015, He *et al.* introduced Residual Neural Network (ResNet) [17] which contains a novel technique called "skip connections". Thanks to this new structure, the network architectures are able to go into very deep with lower complexity than VGGNet. ResNets have three commonly used variants with 50, 101, 152 layers respectively. In this paper, the *conv*1 to *conv*5 and *pool*1 features (ResNets do not have pooling layers for the last four blocks) are investigated in image classification and retrieval tasks, the *conv*1 to *conv*4 and *pool*1 (RPN of faster RCNN is built on top of *conv*4 feature of ResNets, so *conv*5 is not include here) features are involved in image object detection task. To broadly investigate the features of three variants of ResNets while economizing the implementation difficulty, we apply ResNet-152 for image classification following [16], ResNet-50 for image retrieval following [23], and ResNet-101 for image object detection following [4].

*4.1.3 Configurations for compression.* The proposed video codec based lossy compression framework in Section 3.1 is applied in the experiments. Specifically, for the pre-quantization and pre-dequantization modules, the intermediate deep features are quantized/dequantized with a simple logarithmic sampling method:

$$X_{quant} = round\left(\frac{log(X - min(X) + 1)}{max(log(X - min(X) + 1))} \cdot (2^8 - 1)\right) \quad (5)$$

$$X_{dequant} = 2^{\frac{X_{quant} \cdot max(log_2(X - min(X) + 1))}{2^8 - 1}} + min(X) - 1. \quad (6)$$

For the repack module, the size of feature maps is extend to be the integral multiple of 8 by padding after the last array element along each dimension with repeating border elements. The order of the feature map channels are kept the same, as only intra coding will be applied subsequently. As to the video encoder/decoder modules, the reference software (HM16.12) of HEVC Range extension (REext) is employed in the experiments. The compression is performed with four quantization parameter (QP) values, i.e., [12, 22, 32, 42].

## 4.2 Results

The intermediate deep features are firstly extracted by neural networks, then passed to the feature encoder to generate compact bitstreams. The compression rate is subsequently calculated with the volume of original intermediate deep features and the corresponding bitstreams by Equation 1. As to the fidelity evaluation, the reconstructed features are passed to their birth-layer of the corresponding neural network to infer the network outputs, which will be compared with pristine outputs to evaluate the information loss of the lossy compression methods by the proposed metrics described in Section 3.2. The exhaustive results are listed in Table 3.

Comparing with lossless compression results reported in [5], we can see that lossy deep feature compression methods have more potential to compress the feature data into smaller volume than the lossless methods. In the extreme case, i.e. ResNet *conv*4 feature on retrieval dataset can reach over 500x compression ratio at *QP*44, while the lossless methods can only reach 2-5x. However, greater compression ratio will result in greater information loss. For each feature type, the fidelity value decreases as the QP value rises. Looking into the table, it can be also observed that *QP*22 can generally provide high fidelity and fair compression ratio in the meantime. Moreover, upper layer features, like *conv*4 to *pool*5, are generally more robust to heavy compression. It is a great character for practical implementation of intermediate feature transmission, since the high layer features can largely save the computing load while provide great usability at the cloud end, as mentioned in Table 2.

*4.2.1 Visual signal v.s. deep feature.* When transmitting intermediate deep features instead of visual signals, one may concern that if compressed intermediate features are with larger volume than video/image streams, then feature transmission may lose its sense. In Figure 5, we compare the compressed features with the corresponding input visual signal size. To ensure a creditable comparison, we choose the compressed feature volume under *QP*22, as such QP value can provide high fidelity and a fair compression ratio. From the figure, we can observe that upper layer features are generally with smaller compressed size than the lower layer features. All the

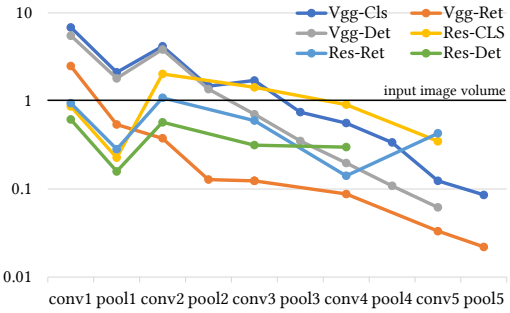



**Figure 5: Data volume comparison for compressed intermediate deep features at *QP*22. The data volumes are scaled with the corresponding input image size. One unit is equivalent to the data volume of input image. *Cls*, *Ret* and *Det* denotes to the features extracted on Classification, Retrieval and Detection datasets, respectively.**

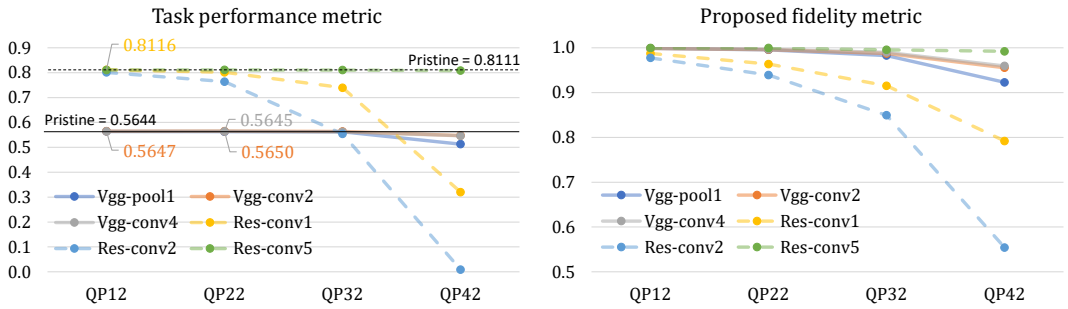
features after *conv*4 can surpass the original visual signal. Considering that the input image can also be compressed to around 10x smaller in common case, we can see there are still quite a few features can reach over 0.1 equivalent data volume of input images. In view of this, some types of feature can provide smaller bandwidth cost than the video/image bit streams. So, it makes sense to transmit intermediate deep features in these cases. Furthermore, if coding the features with larger QP values or more advanced intermediate deep feature compression methods in the future, the feature bitstream can surpass the video/image bitstream in more cases.

Moreover, even if some feature types can not be compressed to smaller size than the video/image bitstreams, it is also meaningful to transmit the features. As the computing resource costs more than the bandwidth in many application scenarios, distributing the computing load to edge-side devices with feature transmission strategy will be meaningful.

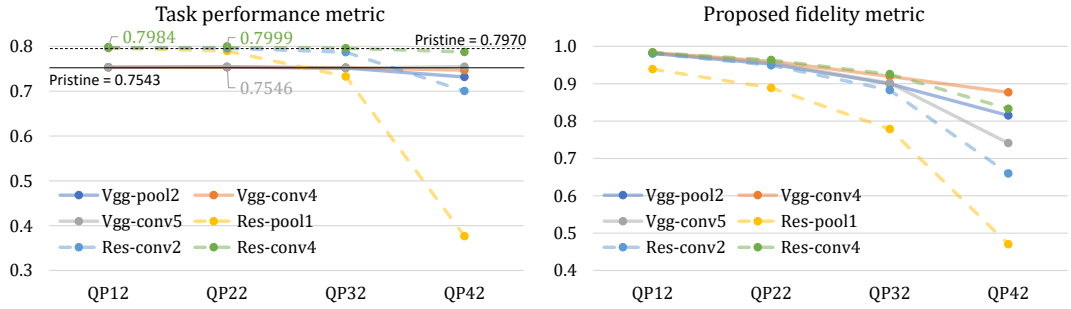
*4.2.2 Metric comparison.* To check the effectiveness of our proposed fidelity metrics, we compare them with the task performance metrics. As shown in Figure 6, such two types of metrics are compared on three selected feature types for each network architecture on retrieval and detection tasks. For the task performance tasks, to evaluate the information loss, the performance values on reconstructed features should be compared with the pristine performance values. As the performance values may vary subject to difference models, datasets and tasks, it is not a straightforward way to measure the information loss. Moreover, although task performance metrics and our proposed fidelity metrics generally share similar trend as QP changes, some task performance values are counter-intuitive. As marked in Figure 6, reconstructed ResNet *conv*1 at *QP*12 and VGGNet *conv*2 at *QP*12&22 on retrieval task, ResNet *conv*4 at *QP*22&32 and VGGNet *conv*5 at *QP*32 can even produce higher performance values than the pristine input. It is contrary to the fact that lossy compression will introduce information loss and higher QP will make the information loss greater. So we believe that our proposed metrics can better evaluate the performance of coding methods which should mainly consider how the reconstructed feature is fidelity to the original feature signal.

**Table 3: Lossy feature compression results (Comp.Rate|Fidelity).**

| Feat. | Classification | | | | Retrieval | | | | Detection | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | QP12 | QP22 | QP32 | QP42 | QP12 | QP22 | QP32 | QP42 | QP12 | QP22 | QP32 | QP42 |
| VGGNet | | | | | | | | | | | | |
| *conv1* | 0.116 \| 0.996 | 0.080 \| 0.985 | 0.048 \| 0.955 | 0.020 \| 0.839 | 0.070 \| 0.999 | 0.041 \| 0.997 | 0.019 \| 0.989 | 0.006 \| 0.955 | 0.096 \| 0.982 | 0.065 \| 0.954 | 0.037 \| 0.913 | 0.013 \| 0.850 |
| *pool1* | 0.145 \| 0.994 | 0.099 \| 0.984 | 0.057 \| 0.914 | 0.023 \| 0.693 | 0.071 \| 0.999 | 0.039 \| 0.996 | 0.017 \| 0.983 | 0.005 \| 0.923 | 0.127 \| 0.977 | 0.085 \| 0.942 | 0.047 \| 0.893 | 0.018 \| 0.820 |
| *conv2* | 0.130 \| 0.992 | 0.098 \| 0.972 | 0.066 \| 0.952 | 0.035 \| 0.790 | 0.089 \| 0.999 | 0.069 \| 0.996 | 0.050 \| 0.987 | 0.027 \| 0.955 | 0.121 \| 0.980 | 0.090 \| 0.950 | 0.059 \| 0.907 | 0.030 \| 0.858 |
| *pool2* | 0.185 \| 0.995 | 0.138 \| 0.982 | 0.090 \| 0.947 | 0.047 \| 0.745 | 0.157 \| 0.999 | 0.119 \| 0.997 | 0.079 \| 0.990 | 0.037 \| 0.945 | 0.172 \| 0.981 | 0.128 \| 0.953 | 0.082 \| 0.900 | 0.040 \| 0.815 |
| *conv3* | 0.102 \| 0.995 | 0.080 \| 0.986 | 0.057 \| 0.960 | 0.034 \| 0.840 | 0.112 \| 0.999 | 0.089 \| 0.998 | 0.070 \| 0.993 | 0.048 \| 0.976 | 0.040 \| 0.981 | 0.033 \| 0.954 | 0.025 \| 0.912 | 0.015 \| 0.845 |
| *pool3* | 0.179 \| 0.989 | 0.140 \| 0.981 | 0.102 \| 0.955 | 0.063 \| 0.819 | – | – | – | – | 0.080 \| 0.982 | 0.066 \| 0.955 | 0.050 \| 0.908 | 0.032 \| 0.826 |
| *conv4* | 0.065 \| 0.992 | 0.053 \| 0.984 | 0.041 \| 0.967 | 0.028 \| 0.865 | 0.088 \| 0.999 | 0.070 \| 0.997 | 0.057 \| 0.990 | 0.043 \| 0.959 | 0.022 \| 0.983 | 0.019 \| 0.960 | 0.014 \| 0.920 | 0.008 \| 0.877 |
| *pool4* | 0.160 \| 0.992 | 0.127 \| 0.974 | 0.097 \| 0.969 | 0.065 \| 0.864 | – | – | – | – | 0.049 \| 0.984 | 0.041 \| 0.960 | 0.031 \| 0.914 | 0.019 \| 0.847 |
| *conv5* | 0.059 \| 0.997 | 0.046 \| 0.989 | 0.037 \| 0.969 | 0.023 \| 0.920 | 0.049 \| 0.998 | 0.041 \| 0.995 | 0.036 \| 0.984 | 0.030 \| 0.952 | 0.031 \| 0.983 | 0.023 \| 0.956 | 0.014 \| 0.902 | 0.005 \| 0.741 |
| *pool5* | 0.162 \| 0.995 | 0.129 \| 0.986 | 0.106 \| 0.967 | 0.075 \| 0.908 | 0.243 \| 0.998 | 0.200 \| 0.996 | 0.173 \| 0.986 | 0.146 \| 0.960 | – | – | – | – |
| ResNet | | | | | | | | | | | | |
| *conv1* | 0.070 \| 0.982 | 0.041 \| 0.935 | 0.019 \| 0.818 | 0.005 \| 0.356 | 0.044 \| 0.987 | 0.018 \| 0.964 | 0.006 \| 0.915 | 0.001 \| 0.792 | 0.056 \| 0.948 | 0.029 \| 0.915 | 0.011 \| 0.872 | 0.002 \| 0.713 |
| *pool1* | 0.074 \| 0.979 | 0.043 \| 0.937 | 0.017 \| 0.732 | 0.004 \| 0.087 | 0.055 \| 0.989 | 0.025 \| 0.963 | 0.009 \| 0.900 | 0.002 \| 0.720 | 0.061 \| 0.939 | 0.030 \| 0.889 | 0.010 \| 0.779 | 0.002 \| 0.470 |
| *conv2* | 0.139 \| 0.995 | 0.095 \| 0.986 | 0.052 \| 0.957 | 0.014 \| 0.765 | 0.061 \| 0.978 | 0.027 \| 0.939 | 0.005 \| 0.849 | 0.001 \| 0.554 | 0.154 \| 0.981 | 0.107 \| 0.949 | 0.061 \| 0.883 | 0.016 \| 0.660 |
| *conv3* | 0.184 \| 0.996 | 0.134 \| 0.989 | 0.083 \| 0.973 | 0.028 \| 0.854 | 0.071 \| 0.996 | 0.041 \| 0.986 | 0.012 \| 0.930 | 0.003 \| 0.551 | 0.162 \| 0.989 | 0.118 \| 0.971 | 0.071 \| 0.922 | 0.021 \| 0.684 |
| *conv4* | 0.231 \| 0.997 | 0.170 \| 0.992 | 0.101 \| 0.977 | 0.034 \| 0.932 | 0.050 \| 0.998 | 0.035 \| 0.997 | 0.022 \| 0.978 | 0.012 \| 0.799 | 0.082 \| 0.984 | 0.056 \| 0.964 | 0.029 \| 0.926 | 0.008 \| 0.833 |
| *conv5* | 0.168 \| 1.000 | 0.131 \| 0.998 | 0.100 \| 0.988 | 0.063 \| 0.961 | 0.057 \| 0.999 | 0.040 \| 0.999 | 0.023 \| 0.996 | 0.014 \| 0.992 | – | – | – | – |




(a) Metric comparison on Retrieval




(b) Metric comparison on Detection

**Figure 6: Comparisons between task performance metrics and the proposed fidelity metrics.**

## 5 CONCLUSIONS

This paper presented a new strategy which compresses and transmits intermediate deep features instead of visual signal or ultimately utilized features in cloud-based visual analysis. The proposed strategy helps reducing the computing load at the cloud end while maintaining the availability of various visual analysis applications, such that better trade-off can be achieved in terms of the computational load, communicational cost and generalization capability. As the first attempt to this problem, we also developed a video codec based lossy compression framework and evaluation metrics for intermediate deep feature compression. Comprehensive experimental results demonstrated the effectiveness of our proposed methods and the feasiblity of the proposed data transmission strategy.

**Acknowledgement** This research is supported by the NTU-PKU Joint Research Institute, a collaboration between the Nanyang Technological University (NTU), Singapore, and Peking University (PKU), China, which is sponsored by a donation from the Ng Teng Fong Charitable Foundation. The research work was done at the Rapid-Rich Object Search (ROSE) Lab at NTU. This work is also supported in part by Singapore Ministry of Education Tier-2 Fund MOE2016-T2-2-057(S), in part by the National Natural Science Foundation of China under Grant 61661146005 and Grant U1611461, in part by Hong Kong RGC Early Career Scheme 9048122 (CityU 21211018), in part by City University of Hong Kong under Grant 7200539/CS, and in part by the National Research Foundation, Prime Minister's Office, Singapore, through the NRF-NSFC Grant, under Grant NRF2016NRF-NSFC001-098.

## REFERENCES


[1] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. 2016. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316* (2016).

[2] Benjamin Bross, Jianle Chen, and Shan Liu. 2018. Working Draft 3 of Versatile Video Coding. *Joint Video Exploration Team of ITUâĂŘT SG16 WP3 and ISO/IEC JTC1/SC29/WG11, JVET-L1001-v9* (2018).

[3] Vijay Chandrasekhar, Jie Lin, Olivier Morere, Hanlin Goh, and Antoine Veillard. 2016. A practical guide to CNNs and Fisher Vectors for image instance retrieval. *Signal Processing* 128 (2016), 426–439.

[4] Xinlei Chen and Abhinav Gupta. 2017. An Implementation of Faster RCNN with Study for Region Sampling. *arXiv preprint arXiv:1702.02138* (2017).

[5] Zhuo Chen, Weisi Lin, Shiqi Wang, Lingyu Duan, and Alex C Kot. 2018. Intermediate deep feature compression: the next battlefield of intelligent sensing. *arXiv preprint arXiv:1809.06196* (2018).

[6] Hyomin Choi and Ivan V Bajic. 2018. Deep feature compression for collaborative object detection. *arXiv preprint arXiv:1802.03931* (2018).

[7] Hyomin Choi and Ivan V Bajic. 2018. Near-Lossless Deep Feature Compression for Collaborative Intelligence. *arXiv preprint arXiv:1804.09963* (2018).

[8] Samuel Dodge and Lina Karam. 2016. Understanding how image quality affects deep neural networks. In *Quality of Multimedia Experience (QoMEX), 2016 Eighth International Conference on*. IEEE, 1–6.

[9] Ling-Yu Duan, Vijay Chandrasekhar, Jie Chen, Jie Lin, Zhe Wang, Tiejun Huang, Bernd Girod, and Wen Gao. 2016. Overview of the MPEG-CDVS Standard. *IEEE Transactions on Image Processing* 25, 1 (2016), 179–194.

[10] Ling-Yu Duan, Vijay Chandrasekhar, Shiqi Wang, Yihang Lou, Jie Lin, Yan Bai, Tiejun Huang, Alex Chichung Kot, and Wen Gao. 2017. Compact Descriptors for Video Analysis: the Emerging MPEG Standard. *arXiv preprint arXiv:1704.08141* (2017).

[11] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. [n. d.]. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

[12] Akira Fukui, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach. 2016. Multimodal compact bilinear pooling for visual question answering and visual grounding. *arXiv preprint arXiv:1606.01847* (2016).

[13] Ross Girshick. 2015. Fast r-cnn. *arXiv preprint arXiv:1504.08083* (2015).

[14] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. 2016. Region-based convolutional networks for accurate object detection and segmentation. *IEEE transactions on pattern analysis and machine intelligence* 38, 1 (2016), 142–158.

[15] Jiuxiang Gu, Jianfei Cai, Gang Wang, and Tsuhan Chen. 2017. Stack-captioning: Coarse-to-fine learning for image captioning. *arXiv preprint arXiv:1709.03376* (2017).

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. [n. d.]. Deep Residual Learning for Image Recognition. https://github.com/KaimingHe/deep-residual-networks.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 770–778.

[18] Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. 2017. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, Vol. 1. 3.

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[20] Pei Li, Loreto Prieto, Domingo Mery, and Patrick Flynn. 2018. Face Recognition in Low Quality Images: A Survey. *arXiv preprint arXiv:1805.11519* (2018).

[21] Jie Lin, Ling-Yu Duan, Shiqi Wang, Yan Bai, Yihang Lou, Vijay Chandrasekhar, Tiejun Huang, Alex Kot, and Wen Gao. 2017. Hnip: Compact deep invariant representations for video matching, localization, and retrieval. *IEEE Transactions on Multimedia* 19, 9 (2017), 1968–1983.

[22] Hongye Liu, Yonghong Tian, Yaowei Yang, Lu Pang, and Tiejun Huang. 2016. Deep relative distance learning: Tell the difference between similar vehicles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2167–2175.

[23] Yihang Lou, Yan Bai, Jun Liu, Shiqi Wang, and Ling-Yu Duan. 2019. Embedding Adversarial Learning for Vehicle Re-Identification. *IEEE Transactions on Image Processing* (2019).

[24] Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. 2016. Hierarchical question-image co-attention for visual question answering. In *Advances In Neural Information Processing Systems*. 289–297.

[25] Wanli Ouyang and Xiaogang Wang. 2013. Joint deep learning for pedestrian detection. In *Proceedings of the IEEE International Conference on Computer Vision*. 2056–2063.

[26] Rohith Polishetty, Mehdi Roopaei, and Paul Rad. 2016. A next-generation secure cloud-based deep learning license plate recognition for smart cities. In *Machine Learning and Applications (ICMLA), 2016 15th IEEE International Conference on*. IEEE, 286–293.

[27] Joseph Redmon and Ali Farhadi. 2018. YOLOv3: An Incremental Improvement. *arXiv preprint arXiv:1804.02767* (2018).

[28] ALESSANDRO ENRICO CESARE Redondi, Luca Baroffio, Lucio Bianchi, Matteo Cesana, and Marco Tagliasacchi. 2016. Compress-then-analyze vs analyze-then-compress: what is best in visual sensor networks? *IEEE Transactions on Mobile Computing* 15, 12 (2016), 3000–3013.

[29] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*. 91–99.

[30] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. 2015. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision* 115, 3 (2015), 211–252.

[31] K. Simonyan and A. Zisserman. [n. d.]. ILSVRC-2014 model (VGG team) with 16 weight layers. https://gist.github.com/ksimonyan/211839e770f7b538e2d8.

[32] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[33] Gary J Sullivan, Jens Ohm, Woo-Jin Han, and Thomas Wiegand. 2012. Overview of the high efficiency video coding (HEVC) standard. *IEEE Transactions on circuits and systems for video technology* 22, 12 (2012), 1649–1668.

[34] Yi Sun, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang. 2014. Deep learning face representation by joint identification-verification. In *Advances in neural information processing systems*. 1988–1996.

[35] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. 2014. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1701–1708.

[36] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *CVPR*.

[37] Lijun Wang, Wanli Ouyang, Xiaogang Wang, and Huchuan Lu. 2015. Visual tracking with fully convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*. 3119–3127.

[38] Yang Wang, Xuemin Lin, Lin Wu, and Wenjie Zhang. 2017. Effective multi-query expansions: Collaborative deep networks for robust landmark retrieval. *IEEE Transactions on Image Processing* 26, 3 (2017), 1393–1404.

[39] Thomas Wiegand, Gary J Sullivan, Gisle Bjontegaard, and Ajay Luthra. 2003. Overview of the H. 264/AVC video coding standard. *IEEE Transactions on circuits and systems for video technology* 13, 7 (2003), 560–576.

[40] Tong Xiao, Hongsheng Li, Wanli Ouyang, and Xiaogang Wang. 2016. Learning deep feature representations with domain guided dropout for person re-identification. In *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*. IEEE, 1249–1258.

[41] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*. 2048–2057.

[42] Chen Zhuo, Fan Kui, Lin Weisi, Duan Lingyu, Kot Alex, C., and Huang Tiejun. 2018. The Framework and Test Condition for Lossy Compression of Deep Feature Maps. *Audio Video Coding Standard (AVS) document AI M1061* (2018).