Sensitivity-Aware Bit Allocation for Intermediate Deep Feature Compression

Yuzhang Hu, Sifeng Xia, Wenhan Yang and Jiaying Liu*
Wangxuan Institute of Computer Technology, Peking University

***Abstract*—In this paper, we focus on compressing and transmitting deep intermediate features to support the prosperous applications at the cloud side efficiently, and propose a sensitivity-aware bit allocation algorithm for the deep intermediate feature compression. Considering that different channels' contributions to the final inference result of the deep learning model might differ a lot, we design a channel-wise bit allocation mechanism to maintain the accuracy while trying to reduce the bit-rate cost. The algorithm consists of two passes. In the first pass, only one channel is exposed to compression degradation while other channels are kept as the original ones in order to test this channel's sensitivity to the compression degradation. This process will be repeated until all channels' sensitivity is obtained. Then, in the second pass, bits allocated to each channel will be automatically decided according to the sensitivity obtained in the first pass to make sure that the channel with higher sensitivity can be allocated with more bits to maintain accuracy as much as possible. With the well-designed algorithm, our method surpasses state-of-the-art compression tools with on average 6.4% BD-rate saving.**

***Index Terms*—intermediate feature compression, bit allocation, video coding, sensitivity measurement, deep learning**

I. Introduction

With the prosperous development of deep learning, it has shown excellent capacities in almost every computer vision fields, including image classification [1], object detection [2], image restoration [3]. The efficiency of deep learning has been universally admitted while there remain some defects during its more extensive application. Due to the tremendous demand for computing resources, the neural networks are always powered by GPUs on the cloud with the original images captured by the user end and transmitted through the network. As a result, there comes some issues. First, transmitting original images might result in the disclosure of private information as the cloud end might frequently be attacked by hackers. Besides, with the explosive growth of image and video data, it brings great computation pressure on the cloud side and the service latency would thus rise to a high level.

Intermediate feature compression can bring a new solution to the above problems. Neural networks consist of consecutively connected layers. Each of these layers stands for a specific kind of computation, which will be performed in turn and the order is consistent with the sequence of layers. After the computation of a layer, its corresponding feature is obtained and can be viewed as the input of subsequent network layers. By compressing and transmitting the intermediate layer's feature (intermediate feature), the above issues can be addressed to a certain extent. First, the computation before the current intermediate feature can be done at the user end, which reduces the burden of the cloud end. Besides, some private information can be hidden via encoding the images/videos into features while still keep the normal working of the neural network. What is more, the compressed feature can be reused for different tasks and the computation before the current intermediate feature is only performed once to save the computational cost. Therefore, intermediate feature compression is of high application value.

In [4], Chen *et al.* make the first attempt to formulate a pipeline for both lossless and lossy feature compression. Taking storage and transmitting costs into consideration, lossy feature compression is more feasible. The to-be-compressed feature will be first quantized from float data to integer data. Then, considering that feature is 3-D data, which shows similarity to video data, traditional video codecs like High Efficiency Video Coding (HEVC) [5] are then used to compress the quantized feature. However, different from video compression, the reference result obtained based on the reconstructed feature measures the quality of the compression algorithm for intermediate feature compression. Take the image classification task as an example, if the degradation caused by the compression process on the reconstructed feature makes the classification result change to another category, this compression algorithm is not a good one. In [4], each channel of the to-be-compressed feature is viewed as a frame and all channels are set with the same quantization parameter (QP), which means there is no difference between the bits allocated for each channel. Unfortunately, it shows larger information divergence among channels compared to the one between video frames. That is, some channel plays a critical role in the final calculation of the neural network while others show a smooth pattern and contribute little to the subsequent computation. Lossy compression will result in loss of information which can be reduced by more allocated bits. If all channels are set to the same QPs, the bits allocated to the unimportant channels are a waste while the bits allocated to the critical channels are limited. This imbalance will lead to greater deviation in the subsequent calculation with the reconstructed feature.

In the traditional video codecs, bit allocation is an important tool to improve the quality of the reconstructed video while

*Corresponding Author. This work was supported by National Natural Science Foundation of China under contract No.61772043, and Beijing Natural Science Foundation under contract No. 4192025 and No.L182002.

978-1-7281-8068-7/20/$31.00 ©2020 IEEE

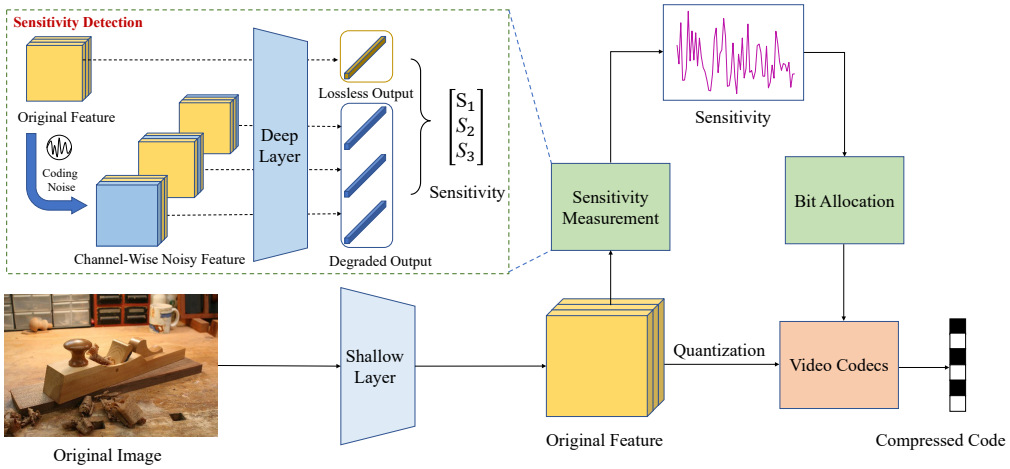


Fig. 1. Pipeline of the Sensitivity-Aware Bit Allocation for intermediate deep feature compression. For simplicity, the to-be-compressed feature consists of 3 channels. During the sensitivity measurement, the blue channel is degraded with the compression noise while others with yellow color are kept as the original ones. All channels will be processed as described above to obtain the channel-wise noisy features, which will be fed to deeper layers to calculate the degraded output of the neural network. The mean squared error between the lossless and degraded output is regarded as the sensitivity. Finally, bit allocation between all channels will be done based on the evaluated sensitivity to compress the feature with video codecs.

keep the total bits at a low level. Frames are divided into key/non-key frames. The former will be allocated with more bits and therefore have better reconstructed quality, which makes it serve as a better reference for the compression on the non-key frames. In this way, the overall reconstructed quality is improved while keeping the extra bits at a low level. Motivated by the bit allocation method in traditional video codecs, we design a sensitivity-aware bit allocation method for intermediate feature compression as shown in Fig. 1, which consists of the process of sensitivity measurement and bit allocation. Specifically, we define and evaluate each channel's sensitivity to the compression noise and perform bit allocation based on each channel's sensitivity to allocate more bits to the channels which are critical to the final reference result. We evaluate the proposed feature compression algorithm on two tasks, *i.e.* image classification and semantic segmentation, to show the superiority of the sensitivity-aware bit allocation method for intermediate feature compression.

II. Sensitivity-Aware Bit Allocation for Intermediate Feature Compression

A. Problem Formulation of Bit Allocation for Feature Compression

Due to the information loss caused by lossy compression, there might be a deviation for the final inference result based on the reconstructed feature. More bits can reduce this information loss and make the inference result more accurate. In other words, to improve inference accuracy as much as possible with the smallest extra bits is the goal of bit allocation for feature compressing. Let θ denote the bit allocation strategy for feature X, then the reconstructed feature denoted as $X^{'}$ can be obtained as follows:

$$X^{'} = V(X, \theta), \tag{1}$$

where V stands for the video codecs. The optimal bit allocation strategy can be represented as follows:

$$\underset{\theta}{argmin}(L(X^{'}) + \lambda B(X, \theta)), \tag{2}$$

where L is the metric to measure the inference accuracy drop with the reconstructed feature $X^{'}$, B is the total bits of the compressed result, and λ defines the weights between these two items.

B. Sensitivity Measurement

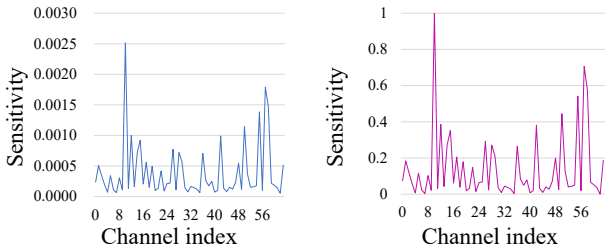
Reconstructed data with lossy compression is different from the original one due to some lossy operations like quantization. We regard this kind of information loss as the compression noise applied to the original data. For the to-be-compressed feature, each channel might show the unique characteristic to the compression noise. For some channels, slight compression noise would lead to great deviation of the final inference result but others just show the opposite property. We regard this property as each channel's sensitivity to the compression noise and evaluate it before the bit allocation step.

For the intermediate layer's feature X of a neural network like VGG [1] or ResNet [6], the sensitivity measurement process will be done as follows. First, the original feature X which has C channels will be fed to the subsequent layers to calculate the lossless output out_{raw}. Next, there will be totally C iterations. At the i-*th* iteration, the i-*th* channel will be compressed alone while others are kept as original. We regard this feature as the i-*th* channel-wise noisy feature and will be fed to the subsequent layers to get the degraded output $out_{(i)}$. The Mean Squared Error (MSE) between out_{raw} and $out_{(i)}$ will be regarded as the i-*th* channel's sensitivity to the compression noise.

Algorithm 1 Bit Allocation for Feature Compression

Input: Original Feature $\mathbf{X} \in R^{C\times H\times W}$,
Base QP QP_{base}, Bit Allocation Magnitude QP_{range},
Subsequent Layers Net, Video Codecs V,
Quantization Function Q,
Dequantization Function DQ
Output: Compressed feature $\widetilde{\mathbf{X}}$

$out_{raw} \leftarrow Net(\mathbf{X})$
for each channel x_i of $\mathbf{X}$, **do**
  $\widetilde{x_i} \leftarrow DQ(V(Q(x_i), QP_{base}))$
  // $x_i \in R^{1\times H\times W}$
  $\overline{\mathbf{X}} \leftarrow copy(\mathbf{X})$
  i-*th* channel of $\overline{\mathbf{X}} \leftarrow \widetilde{x_i}$
  $out_{(i)} \leftarrow Net(\overline{\mathbf{X}})$
  $\mathbf{S_i} \leftarrow$ MSE $(out_{(i)}, out_{raw})$
  // All channels' sensitivity $\mathbf{S} \in R^C$
$\widetilde{\mathbf{S}} \leftarrow \frac{\mathbf{S}-\min(\mathbf{S})}{\max(\mathbf{S})-\min(\mathbf{S})}$
$\mathbf{QP} \leftarrow QP_{base} - round(QP_{range} \cdot \widetilde{\mathbf{S}})$
// $\mathbf{QP} \in Q^C$ stands for the quality parameter for
// each channel during compression
$\widetilde{\mathbf{X}} \leftarrow V(Q(\mathbf{X}), \mathbf{QP})$
return $\widetilde{\mathbf{X}}$



(a) Raw sensitivity (b) Relative sensitivity

Fig. 2. Visulization of the sensitivity.

C. Bit Allocation

After the sensitivity measurement, all channels' sensitivity to the compression noise $\mathbf{S} \in R^C$ is obtained. As Fig. 2 (a) shows, there remains an unregular distribution in the raw sensitivity. Specifically, the sensitivity gap between different channels is huge, which makes it difficult to unify the bit allocation among all channels. As a result, we propose to normalize raw sensitivity as follows:

$$\widetilde{\mathbf{S}} = \frac{\mathbf{S} - \min(\mathbf{S})}{\max(\mathbf{S}) - \min(\mathbf{S})}. \tag{3}$$

The normalized sensitivity is regarded as the relative sensitivity $\widetilde{\mathbf{S}}$ as shown in Fig. 2 (b) and the QP for each channel will be decided according to $\widetilde{\mathbf{S}}$. The hyper parameter QP_{base} will be set as an approximate target rate level and QP_{range} is used to control the magnitude of bit allocation. Then the i-*th* channel's QP is set to $QP_{base} - round(QP_{range} \cdot \widetilde{\mathbf{S}_\mathbf{i}})$. For the video codecs, smaller QP means more bits and thus the channel with higher sensitivity will suffer from less information loss caused by lossy compression. Finally, the quantized feature will be compressed by video codecs with **QP** calculated for each channel. The detailed algorithm is shown in Algorithm 1.

III. Experiment

A. Experiment Setting

We choose HM-16.12 as the video codecs and compress the quantized feature under the all intra coding configuration. During the compression process, each quantized channel is regarded as a grey frame so the color space for video codecs is set to 400.

VGG [1] and ResNet [6] are two most widely used neural networks and we compress the feature of their first 2 shallow layers which are named following [4] with the proposed method. We test the tasks of *image classification* on VGG16 and *semantic segmentation* on ResNet50 as shown in Table I.

TABLE I
Tested layers of VGG and ResNet

Task	*Image Classification*	*Semantic Segmentation*
Backbone	VGG16	ResNet50
Layer	Conv1/Pool1	Conv1/Pool2

The most important metric to evaluate a feature compression algorithm are the compression rate and fidelity. The compression rate stands for the ratio of bitstream size to uncompressed feature size. The fidelity measures the inference deviation caused by lossy compression. The compression method proposed in [4] which sets the same QP for all channels under all intra configuration is chosen as the anchor. For short, we call it unified QP method.

More details will be described in the following section.

B. Implementation Details

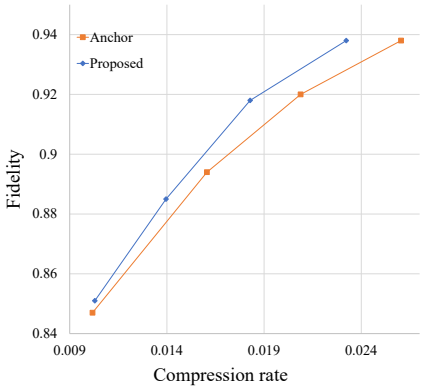
We use the pretrained weights provided by PyTorch [7] to initialize the neural network of the to-be-compressed feature. In the sensitivity measurement step, the channel-wise noisy feature will be fed to the subsequent layers except for the final classifier to calculate the degraded output.

We choose VGG16 on the task of image classification. 1000 images of ImageNet [8] collected by *Chen et al.* [4] are used as the testing dataset. Image classification is the image-level task so we directly use the fidelity defined in [4] to measure the inference deviation caused by lossy compression as follows:

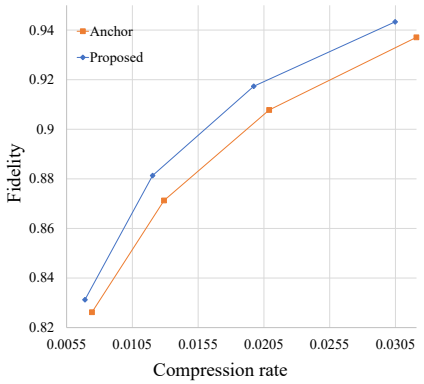
$$fid(M, \widetilde{M}) = \begin{cases} 1, & argmax(M) = argmax(\widetilde{M}), \\ 0, & else, \end{cases} \tag{4}$$

where M and $\widetilde{M}$ are the vectors obtained by the fully connected classifier with original and reconstructed features respectively.

We choose FCN [9] with the ResNet50 as the backbone on the task of semantic segmentation on PASCAL 2007 [10]. Different from image classification, semantic segmentation is



(a) VGG16 (Image Classification)



(b) ResNet50 (Semantic Segmentation)

Fig. 3. The Rate-Fidelity Curve of layer Conv1 for VGG and ResNet.

a pixel-wise classification task. The fidelity of a reconstructed feature is defined as the average fidelity of all pixels as follows:

$$fid(M, \widetilde{M}) = \frac{1}{H \cdot W} \sum_{x=1}^{H} \sum_{y=1}^{W} fid(M(x,y), \widetilde{M}(x,y)), \quad (5)$$

where H and W are the height and width of the input image. $M(x,y)$ and $\widetilde{M}(x,y)$ are the classification vectors of the pixel at the location (x,y) of the semantic maps obtained with original and reconstructed features respectively.

For image classification task, the QP_{base} parameters are set to 36, 38, 40, 42. For image semantic segmentation task, we find that the fidelity will drop severely for Conv1 with high QPs so we set the QP_{base} parameters to 14, 18, 22, 26 while there is less impact with high QPs for Pool2 so the QP_{base} for this layer is set to 36, 38, 40, 42. The QP_{range} parameter is always fixed to 5.

TABLE II
RATE REDUCTION OF THE PROPOSED METHOD

Layer	*Image Classification*	*Semantic Segmentation*
Conv1	-6.3%	-13.2%
Pool1	-3.2%	-
Pool2	-	-3.0%

C. Experiment Results

The overall performance is shown in Table II. By allocating more bits to the channels which are more sensitive to the compression noise, the inference deviation caused by lossy compression can be suppressed with as less bits as possible. The reconstructed feature obtained by the proposed method can improve the accuracy of both image-level and pixel-level tasks. Specifically, 13.2% bitstream can be saved for the layer Conv1 of ReNet50 on the task of semantic segmentation.

The rate-fidelity curves of Conv1 layer of VGG16 and ResNet50 are shown in Fig. 3. It can be observed that our method shows better compression efficiency compared with the unified QP method.

IV. CONCLUSION

In this paper, we explore the effect of the compression noise on different feature channels and propose a sensitivity-aware bit allocation algorithm for intermediate feature compression. By simulating the inference deviation caused by lossy compression in a channel-wise way, we evaluate the sensitivity of each channel to the compression noise and allocate more bits to the more sensitive ones. Owe to this design, the valid information which plays an important role in the subsequent calculation will suffer from less degradation and thus keep the final inference result more accurate, *i.e.* consistent with the original one while keeping the total storage cost as low as possible.

REFERENCES

[1] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

[2] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, pp. 91–99, 2015.

[3] M. Li, J. Liu, X. Sun, and Z. Xiong, "Image/video restoration via multiplanar autoregressive model and low-rank optimization," *ACM Transactions on Multimedia Computing and Communications and and Applications (TOMM)*, vol. 15, no. 4, pp. 1–23, 2019.

[4] Z. Chen, K. Fan, S. Wang, L. Duan, W. Lin, and A. C. Kot, "Toward intelligent sensing: Intermediate deep feature compression," *IEEE Transactions on Image Processing*, vol. 29, pp. 2230–2243, 2020.

[5] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, 2012.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the Computer Vision and Pattern Recognition (CVPR)*, 2016.

[7] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, pp. 8024–8035, 2019.

[8] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[9] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Annals of the History of Computing*, no. 04, pp. 640–651, 2017.

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