CDC: Classification Driven Compression for Bandwidth Efficient Edge-Cloud Collaborative Deep Learning

Yuanrui Dong[1], **Peng Zhao**[1], **Hanqiao Yu**[1], **Cong Zhao**[2*] and **Shusen Yang**[1]
[1]National Engineering Laboratory for Big Data Analytics, Xi'an Jiaotong University, China
[2]Department of Computing, Imperial College London, UK
d2484769248@stu.xjtu.edu.cn, p.zhao@mail.xjtu.edu.cn, yuhanqiao@stu.xjtu.edu.cn, c.zhao@imperial.ac.uk, shusenyang@mail.xjtu.edu.cn

Abstract

The emerging edge-cloud collaborative Deep Learning (DL) paradigm aims at improving the performance of practical DL implementations in terms of cloud bandwidth consumption, response latency, and data privacy preservation. Focusing on bandwidth efficient edge-cloud collaborative training of DNN-based classifiers, we present CDC, a Classification Driven Compression framework that reduces bandwidth consumption while preserving classification accuracy of edge-cloud collaborative DL. Specifically, to reduce bandwidth consumption, for resource-limited edge servers, we develop a lightweight autoencoder with a classification guidance for compression with classification driven feature preservation, which allows edges to only upload the latent code of raw data for accurate global training on the Cloud. Additionally, we design an adjustable quantization scheme adaptively pursuing the tradeoff between bandwidth consumption and classification accuracy under different network conditions, where only fine-tuning is required for rapid compression ratio adjustment. Results of extensive experiments demonstrate that, compared with DNN training with raw data, CDC consumes 14.9× less bandwidth with an accuracy loss no more than 1.06%, and compared with DNN training with data compressed by AE without guidance, CDC introduces at least 100% lower accuracy loss.

1 Introduction

Recently, the emerging paradigm of collaborative Edge Intelligence (EI) rapidly draws significant interests from both academia [Zhou *et al.*, 2019; Chen and Ran, 2019] and industry[1,2], where the deployment of various Artificial Intelligence (AI) applications is pushed from mega-scale cloud datacenters to heterogeneous devices at network edges closer to explosive end data [Cisco, 2016]. Confronting *intensive computations on the Cloud* and *expensive end data uploading* that are inherently controversial in conventional cloud-based AI deployments, EI has been demonstrated to be promising in reducing cloud bandwidth consumption and response latency, as well as preserving data privacy [Zhou *et al.*, 2019; Chen and Ran, 2019].

As the frontier of the latest flourish of AI, Deep Learning (DL) is of extraordinary success. Even so, the EI-driven solution above is indisputably appealing but challenging to the deployment of urgently required DL applications. Conventionally, highly accurate DL models can be centrally trained on the Cloud with abundant resources [Mell *et al.*, 2011], which, however, requires enormous raw data (*e.g.* images and videos) to be uploaded, and induces expensive bandwidth consumptions [Shi and Dustdar, 2016], especially considering skyrocketing end data sources (*e.g.* IoT devices). Alternatively, direct DNN training at edges with easily accessible end data can significantly reduce the cloud bandwidth consumption [Satyanarayan, 2017], but, in practice, it is quite difficult (or even infeasible) to train naturally complex highly accurate models under severe edge resource constraints. Therefore, the *tradeoff between cloud bandwidth consumption and model accuracy* has to be explicitly addressed for effective edge-cloud collaborative DL.

Obviously, one of the most intuitive methods to reduce cloud bandwidth consumption is to compress raw data at edges before uploading them to the Cloud. Generally, Lossy compression [Shaham and Michaeli, 2018] manages to obviously reduce the data size at the cost of losing details, where the accuracy of models trained on such compressed data may be severely impacted. Therefore, it is critical to develop a data compression method that can obviously reduce the data size while selectively preserving valuable details for accurate model training.

Focusing on such an issue, treating DNN-based classification as a use case, we present a Classification Driven Compression (CDC) framework to reduce bandwidth consumption while preserving model accuracy for effective edge-cloud collaborative DNN training. Particularly, for resource limited edge servers, we design a lightweight autoencoder (AE) with a classification guidance for compression with classification driven feature preservation. In addition, we develop an adjustable quantization scheme for the tradeoff between bandwidth consumption and classification accuracy, where

*Corresponding Author

[1]https://azure.microsoft.com/en-us/overview/future-of-cloud

[2]https://cloud.google.com/blog/products/gcp/bringing-intelligence-edge-cloud-iot

only fine-tuning is required for rapid compression ratio adjustment. Contributions of this paper are as follows:

1. We propose CDC for bandwidth efficient edge-cloud collaborative DNN classifier training, where raw data remain at edges, and only small size latent codes are uploaded to the Cloud. To the best of our knowledge, CDC is the first classification driven compression method.
2. We present a classification guidance approach for the edge compression AE to selectively learn representative features for accurate classification, which allows CDC to reduce bandwidth consumption and preserve classification accuracy simultaneously. Besides, an adjustable quantization scheme is developed to rapidly achieve the tradeoff between bandwidth consumption and classification accuracy under different network conditions.
3. We conducted extensive experiments to evaluate CDC's performance, compared with collaborative DNN training with raw data, CDC manages to consume 14.9× less bandwidth while introducing an accuracy loss no more than 1.06%. Besides, compared with DNN training based on data compressed by traditional autoencoder, CDC manages to introduce at least 100% lower accuracy loss with the same compression ratio.

2 Related Work

Deep learning with edge computing. Emerging efforts [Zhou *et al.*, 2019; Chen and Ran, 2019] have been demonstrating that deploying DL models at network edges can bring about significant performance gains in terms of cloud bandwidth saving, inference latency reduction, and privacy preservation, comparing to predominating cloud-based approaches. For instance, [Teerapittayanon *et al.*, 2017; Li *et al.*, 2018b] use early-exit at a proper intermediate DNN layer to reduce inference latency, and [Li *et al.*, 2018a] focuses on online optimization of the exit point. However, in these efforts, DL models are trained on the Cloud with expensive bandwidth consumptions. Some researchers are interested in distributed training and focus on issues caused by model aggregation, *e.g.* bandwidth consumption [Hardy *et al.*, 2017; Tang *et al.*, 2018; Smith *et al.*, 2017], training latency [Tang *et al.*, 2018], stragglers, and fault tolerance [Smith *et al.*, 2017]. However, these efforts do not leverage edge and cloud resources collaboratively.

Autoencoder with auxiliary task. The potential of autoencoder in image compression has already been widely demonstrated [Theis *et al.*, 2017; Zhou *et al.*, 2018; Li *et al.*, 2018c], and many of these are comparable to the best image compression standards in terms of perceptual metrics. However, these efforts are not for resource-constrained edges. Moreover, in the field of multi-task learning, auxiliary tasks have also been explored through the use of hints for neural networks, and the autoencoder is used as an auxiliary task to improve the performance of classification tasks [Liu *et al.*, 2016; Le *et al.*, 2018]. Beyond that, autoencoder can also accept multiple loss functions, as a main task, to learn a more efficient model [Cipolla *et al.*, 2018]. However, these efforts do not focus on compressing data while retaining features that are valuable for the auxiliary task.

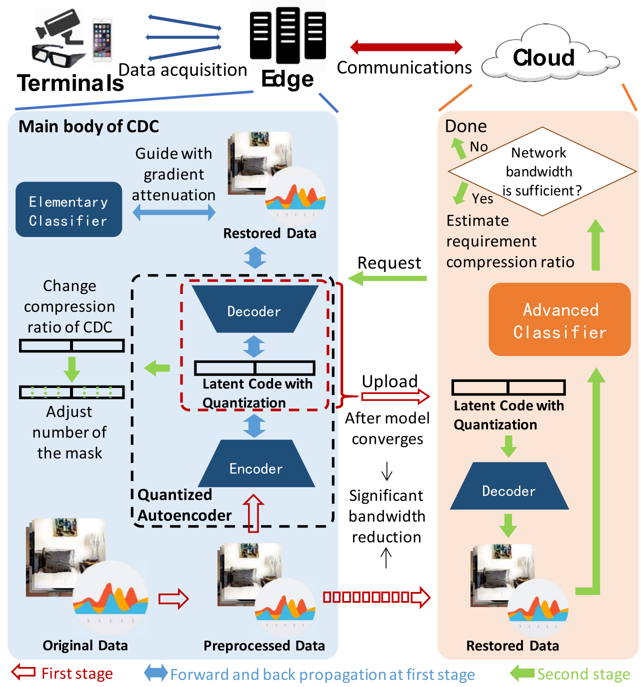


Figure 1: Edge-cloud collaborative DNN training framework. Left: On the Edge. Right: On the Cloud.

3 Classification Driven Compression for Edge-cloud Collaborative DNN Training

3.1 Basic Idea

In this paper, we treat the CNN-based classification as a use case, and focus on the problem of raw data compression that reduces the bandwidth consumption and preserves the classification accuracy simultaneously for effective edge-cloud collaborative DNN training. The architecture of our collaborative DNN training framework is shown in Figure 1.

In our system, the edge server acquires labelled data from various kinds of terminal devices, and a high-accuracy *advanced classifier* (AC, *e.g.* ResNet, ResNeXt, DPN) on the Cloud requires to be trained. Different with existing approaches of large scale DNN training on the Cloud, raw data reside on the edge, where a CDC process is conducted to reduce the bandwidth consumption for the training of AC. In particular, CDC on the edge is based on a *quantized autoencoder* (AE) and an *elementary classifier* (EC). Here, the quantized AE is responsible for compressing raw data into representative features (*i.e.* latent codes) with significantly reduced sizes, where an *adjustable quantizer* is adopted to achieve the tradeoff between data compression ratio and feature preservation ratio. Then, EC focuses on the similar classification task with that of AC, which has a lower accuracy but can be deployed on resource limited edge servers. Integrating EC into CDC aims at providing a 'guidance' for the construction of AE on the edge, which allows AE to effectively compress raw data while preserving information that is critical to the performance of AC on the Cloud. The workflow of our system can be divided into two stages:

Stage 1: For the edge server, AE and EC are jointly trained through gradient descent using local raw data. As shown in Figure 1, the forward and backward propagation of the training loss is illustrated with blue arrows. Specifically, the training of EC is based on the classification loss between restored data from AE and corresponding raw labels. AE is trained with the joint loss of both the reconstruction loss of itself and an attenuated classification loss of EC (which is treated as the 'classification guidance'). After the convergence of AE and EC, raw data are compressed into latent codes, which are uploaded with corresponding labels together with the decoder to the Cloud for the training of AC.

Stage 2: Receiving latent codes and the decoder from the edge server, AC is trained based on restored data on the Cloud. When AC converges, the edge-cloud bandwidth is evaluated: if it is intensely occupied, the training of AC is terminated; else if the bandwidth is sufficient, requests of clearer data are sent to the edge server, together with a proper edge compression ratio estimated by the Cloud. Receiving the request from the Cloud, the edge adjusts its compression ratio through the quantizer of AE and re-enters Stage 1.

3.2 Classification-guided Autoencoder

Overview

As mentioned above, we construct an autoencoder on the edge to compress raw data for the reduction of bandwidth consumption of AC training on the Cloud. To preserve the accuracy of AC trained based on compressed data (*i.e.* latent codes, labels and the decoder), AE and EC are jointly trained on the edge, where the classification loss of EC is used to guide the training of AE. Specifically, the training of EC is a *supervised auxiliary task* that helps the training of AE focusing more on retaining critical features for accurate classifications. In fact, such a classification guidance can be viewed as a certain form of inductive transfer that helps to introduce certain preferences into model training (*i.e.* the successive classification on the Cloud in this paper) by introducing an inductive bias (*i.e.* the classification loss of EC in this paper).

Classification Loss of EC and Reconstruction Loss of AE

Under a supervised learning setting, EC's basic target is to learn a mapping from a vector of inputs $\mathbf{x} \in \mathbb{R}^N$ to a vector of labels $\mathbf{y} \in \mathbb{R}^C$ with the minimum *classification loss* L_C, which is iteratively optimized through the training based on a batch of raw data $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_t, \mathbf{y}_t)$. The classifier output layer has a supervised loss:

$$L_C\left(h_C\left(\mathbf{x}\right), \mathbf{y}\right) = -\frac{1}{t}\sum_{i=1}^{t}\sum_{j=1}^{C}\mathbf{y}_i^{(j)} \ln h_C^{(j)}(\mathbf{x}_i), \qquad (1)$$

where $h_C\left(\mathbf{x}\right)$ denotes the output of the classifier.

Under the same setting, AE's basic target is to learn a mapping from a vector of inputs $\mathbf{x} \in \mathbb{R}^N$ to a vector of reconstructed outputs $\mathbf{x}' \in \mathbb{R}^N$ with the minimum *reconstruction loss* L_A, which is also iteratively optimized through the training based on i.i.d. raw data batches. For a compression AE, the output layer has a reconstruction loss:

$$L_A\left(h_A\left(\mathbf{x}\right), \mathbf{x}\right) = \frac{1}{2t}\sum_{i=1}^{t}\left\|h_A\left(\mathbf{x}_i\right) - \mathbf{x}_i\right\|_2^2, \qquad (2)$$

where $h_A\left(\mathbf{x}\right)$ denotes the output of AE.

In our system, the output of AE is directly fed into EC, and we define the *full loss* as:

$$L_F = L_A\left(h_A\left(\mathbf{x}\right), \mathbf{x}\right) + L_C\left(h_C\left(h_A\left(\mathbf{x}\right)\right), \mathbf{y}\right). \qquad (3)$$

As for the rationality of DNN training based on the full loss above, Le *et al.* [Le *et al.*, 2018] has proved that a classifier trained with a loss function similar to Equation (3) can perform better and be effectively regularized under certain assumptions. It indicates that feature extractions of both the classifier and the AE are compatible to a certain extent, and it is reasonable to train the two models jointly for data compression that preserves classification-related features.

Prioritized Joint Training of AE and EC

For the application of massive amounts of practical data, relatively deeper nonlinear networks with more complex structures (that can still be deployed on resource-limited edges) should be adopted. In this case, we have an observation that AE directly trained with the full loss L_F in Equation (3) may not converge. To address this issue, we design a parameter, the *classification attenuation rate* $\alpha \geq 1$, to adjust the priority of L_C and L_A. Specifically, the full loss is modified as:

$$L_F = L_A\left(h_A\left(\mathbf{x}\right), \mathbf{x}\right) + \frac{1}{\alpha}L_C\left(h_C\left(h_A\left(\mathbf{x}\right)\right), \mathbf{y}\right). \qquad (4)$$

With such a full loss, the impact of classification guidance can be controlled by adjusting α to achieve the convergence of AE. When α is large enough, the impact of classification guidance is negligible.

In our system, AE and EC are jointly trained through gradient decent with the same learning rate but different losses: the gradient of EC training is calculated as:

$$\delta^{EC} = \nabla L_C\left(h_C\left(h_A\left(\mathbf{x}\right)\right), \mathbf{y}\right), \qquad (5)$$

and the gradient of AE training is calculated as:

$$\delta^{AE} = \nabla L_A\left(h_A\left(\mathbf{x}\right), \mathbf{x}\right) + \frac{1}{\alpha}\nabla L_C\left(h_C\left(h_A\left(\mathbf{x}\right)\right), \mathbf{y}\right). \qquad (6)$$

Based on the above design, the impact of classification guidance on the training of AE can be controlled by adjusting the attenuation rate α: a lower α will preserve more critical data features for high-accuracy classification in the constructed AE, which, however, will lead to a more obvious impact on the convergence of AE. Besides, attenuation rate α also significantly affects the visual result of restored data.

3.3 Compression Based on Quantization

To achieve the tradeoff between bandwidth consumption and classification accuracy in edge-cloud collaborative DNN training, we construct a compression autoencoder with an adjustable quantizer for the edge server. In particular, based on the stochastic quantization method proposed by Theis *et*

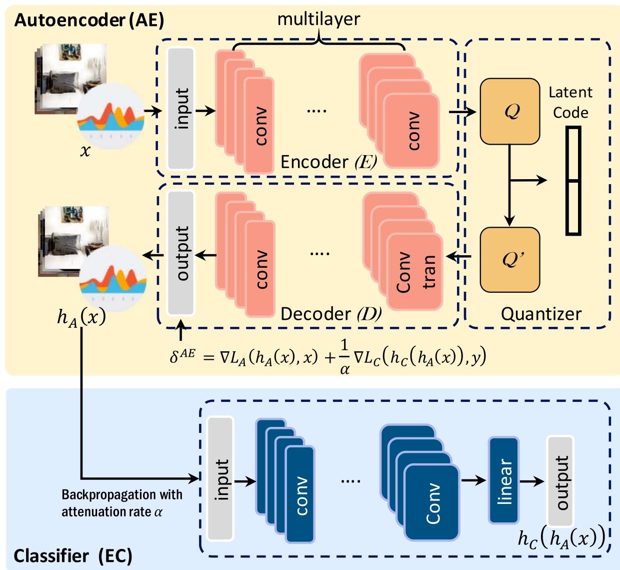


Figure 2: The architecture of CDC model. One-dimensional convolution for audios, two-dimensional for images. Top: The autoencoder with a quantization module. Except for the last convolution of E and D, each convolution and transposed convolution is followed by a leakyReLU activation function. Mean squared error is used as a measure of distortion during training. Bottom: The elementary classifier with convolutions followed by ReLU activation functions, and is interspersed with some pooling layers.

al. [Theis *et al.*, 2017], we design a lightweight compression autoencoder, where a quantizer with an adjustable mask number m is integrated.

As shown in Figure 2, our compression autoencoder contains three major components: the encoder E, the decoder D, and the quantizer Q and Q'. Their definitions (as specifically designed mappings) are as follows:

$$E : \mathbb{R}^N \to (-1,\, 1)^M, \quad D : (-1,\, 1)^M \to \mathbb{R}^N,$$
$$Q : (-1,\, 1)^M \to \left[-2^{4-m} + 1, \ldots, -1, 0, 1, \ldots, 2^{4-m}\right]^M,$$
$$Q' : \left[-2^{4-m} + 1, \ldots, -1, 0, 1, \ldots, 2^{4-m}\right]^M \to (-1,\, 1)^M.$$

In our compression autoencoder, E maps N-dimensional input data into M-dimensional latent codes, where the range of each output dimension is $(-1,\, 1)$. Correspondingly, D is responsible for mapping latent codes into restored data. In addition, Q maps latent codes from E into M-dimensional tuples, where each dimension is an integer within the range of $[-2^{4-m} + 1, 2^{4-m}]$ that can be efficiently stored with bits. Here, m represents the mask number of Q. Correspondingly, Q' restores quantized tuples as specific intervals containing pre-quantized values.

The stochastic quantization process with a uniformly distributed random noise ε is conducted as follows:

$$Q(x) = \left\lceil x \cdot 2^{4-m} + \varepsilon \right\rceil_{(-2^{4-m},\, 2^{4-m})},$$
$$Q'(x) = (x - 0.5)\,/2^{4-m}.$$

Here, $Q(x)$ represents the process of quantized compression, $Q'(x)$ represents the process of quantized compression and reduction, and $\lceil\,\rceil_{(a,\,b)}$ is the ceiling rounding operation with a and b as lower and upper bounds, respectively. Specifically, we use $\varepsilon \in (-0.5,\, 0.5)$ to constrain the output of E, while the derivative of quantization is replaced with the derivative of the expectation. In this way, we can use the regularization provided by the stochastic quantization to cleanly backpropagate gradients through the quantizer, reduce the over-fitting caused by the fluctuation of quantized values, and achieve a better generalization capability [Theis *et al.*, 2017].

In the discussion above, the mask number $m \in \{0, 1, 2, 3, 4\}$ is treated as a *compression/restoration hyper-parameter*, which is used to adjust the ratio of zoom in/out before rounding for quantized compression. Different mask numbers lead to differences in terms of bandwidth consumption and reconstruction error. Specifically, with a lower m, a fewer lower numbers are truncated, and quantized data have a higher precision.

It's worth noting that on the basis of quantization, the entropy coding method, which could further improve the compression ratio, can be easily integrated into our framework.

3.4 Discussions

Applying CDC in inference. It is promising to extend the application of CDC in edge-cloud collaborative DNN inference. Specifically, for CDC, EC on edges can be treated as early exits of inference results [Teerapittayanon *et al.*, 2017; Li *et al.*, 2018a], which can significantly reduce the inference latency, while AC on the Cloud can provide more credible inference results with higher bandwidth consumption and inference latency. Besides, results on edges can also assist the inference on the Cloud, where ensemble learning can be adopted to enhance inference accuracy.

Distributed training. For practical applications, it is promising to combine CDC with other distributed algorithms to achieve efficient training. For example, in the multi-edge scenario, if data on distributed edge nodes are i.i.d., collaborative training can be conducted, where data parallel methods [Li *et al.*, 2014; Abadi *et al.*, 2016; Hardy *et al.*, 2017] can be adopted to improve the training speed on edges. And if data are not i.i.d., the federated learning method [Kairouz *et al.*, 2019] can also be adopted for diverse application.

4 Performance Evaluation

In this section, we conducted extensive experiments to evaluate the performance of CDC on reducing the bandwidth consumption while preserving model accuracy for effective edge-cloud collaborative DNN training. Our experimental methodology and results are as follows.

4.1 System Architecture and Classification Models

In our experiments, we constructed an edge-cloud simulator with a pair of cloud and edge servers. Specifically, we adopted ResNeXt-29 (32x4d) [Xie *et al.*, 2017] as the advanced classifier (AC) on the Cloud for image classification, depth one-dimensional convolutional network for audio classification. An autoencoder (AE) and an elementary classifier (EC) with the structure in Figure 2 were deployed on the edge server. The model training was conducted using `PyTorch`.

4.2 Datasets for Classification

For a comprehensive evaluation, we adopted two different image datasets and two different audio datasets to validate the effectiveness of our approach on practical classification tasks.

The LSUN [Yu *et al.*, 2015] contains 10 classes of images of real-life scenes. Specifically, we selected 10,000 images for each class as the training set on the edge server, and we used the test set provided by LSUN as our test set on the Cloud. **The Vehicles** contains images of 10 classes of vehicles selected from the imagenet dataset [Deng *et al.*, 2009], where 12,000 images were deployed on the edge server as the training set, and 1,000 images were deployed on the Cloud as the test set. **The Urbansound8K** [Salamon *et al.*, 2014] consists of 10 classes of audio data in .wav format. We used the preset folders 1-8 as the training set, folders 9, 10 as the test set. **The FSDD** [Jackson *et al.*, 2018] is an audio dataset with 1,500 recordings in .wav format, corresponding to spoken digits ($0 \sim 9$) from 3 different speakers, where each digit is recited 50 times. We used the official training set and test set division.

4.3 Effectiveness of Classification Guidence

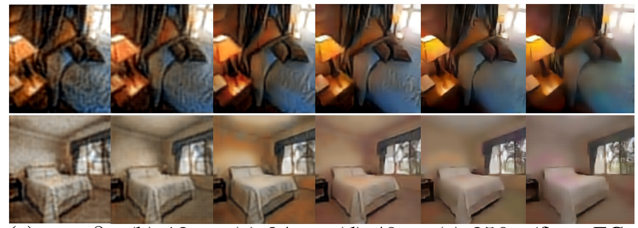
In this set of experiments, we investigated the impact of the classification guidance from the elementary classifier (EC) on the performance of both autoencoder (AE) and advanced classifier (AC). Specifically, we used a fixed quantizer mask number $m = 4$ that corresponds to 1 Bits-Per-Pixel (BPP) image compression ratio, 13.5:1 data compression ratio for Urbansound8K, and 12:1 data compression ratio for FSDD. We developed multiple CDC models (*i.e.* AE + EC) with different attenuation rates and evaluated their capabilities on preserving both raw data features and the classification accuracy of AC. Each CDC model was trained three times, and the one with the highest validating accuracy was selected for the result demonstration. For each CDC model above, AC was trained three times, and, similarly, the one with the highest validating accuracy was selected for the result demonstration.

Feature Preservation

We first discuss the impact of classification guidance attenuation rate on CDC's capability of preserving raw data features. Figure 3 illustrates several image outputs of CDC models with different attenuation rates for a visual comparison. It is obvious that *CDC focuses on raw image details more rigidly when a lower attenuation rate is adopted*, which proves the viability of integrating the training of EC as an auxiliary task for the training of AE. One thing that should be noted is that low attenuation rates lead to severe distortions of raw data since optimization targets of EC and AE are not completely consistent, and insufficiently attenuated guidance will hinder CDC's capability on restoring data.

Classification Accuracy

We now discuss the impact of classification guidance attenuation rate on the classification accuracy of AC. The AC accuracy with CDC models using different attenuation rates is demonstrated in Figure 4, where the AC accuracy with traditional Autoencoder is also depicted for comparison. Obviously, on all adopted datasets, the CDC model with a proper attenuation rate (*i.e.* from our experiments, around 48 for images, 16 for audios) manages to achieve a significantly higher AC accuracy compared with the scenario with traditional Autoencoder. In particular, AC trained on the audios compressed and restored by the traditional autoencoder do not converge, while the data compressed and restored by CDC can effectively support the classification on the Cloud. Such a result indicates that the integration of classification guidance with a proper attenuation rate can obviously enhance the capability of an autoencoder on selectively preserving raw data features that are critical to the training of high-accuracy advanced classifiers on the Cloud.



(a) $\alpha = 8$. (b) 12. (c) 24. (d) 48. (e) 256. (f) no EC.

Figure 3: Visual comparison of CDC compression results with different attenuation rates (with 1 BPP compression ratio).

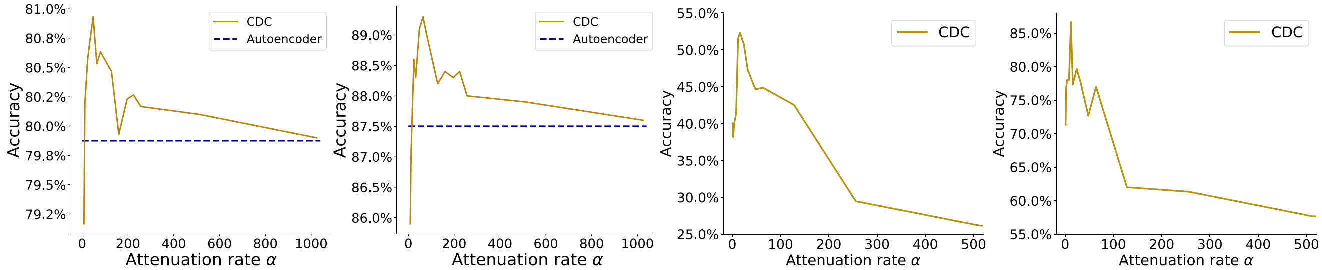
Methods	Accuracy (L / V)(%)	Loss (L / V)(%)
	Accuracy (U / F)(%)	Loss (U / F)(%)
Baseline	81.99 / 90.50	0 / 0
	62.37 / 88.33	0 / 0
AE	79.87 / 87.50	2.12 / 3.00
	12.44 / 10.67	49.93 / 77.66
CDC	80.93 / 89.10	1.06 / 1.40
	52.33 / 86.67	10.04 / 1.66

Table 1: AC accuracy with traditional AE and CDC (with attenuation rate $\alpha = 48$ for image datasets, $\alpha = 16$ for audio). L, V, U and F respectively stand for LSUN, Vehicles, Urbansound and FSDD.

For a further validation, we compared the performance of the optimal CDC model on preserving the AC accuracy with that of traditional Autoencoder. Specifically, we separately trained AC with our preprocessed four datasets without any compression as baselines. Then, with a fixed compression ratio (*i.e.* $m = 4$), AC was trained based on the restored datasets from traditional autoencoder and CDC, respectively. As shown in Table 1, CDC outperforms its comparative under the same compression ratio requirement.

4.4 Tradeoff between Compression Ratio and Classification Accuracy

In this set of experiments, we investigated CDC's capability on achieving the tradeoff between raw data compression (to reduce bandwidth consumption) and the classification accuracy during the edge-cloud collaborative training process. Specifically, with a fixed classification guidance attenuation



(a) On LSUN. (b) On Vehicles. (c) On Urbansound8K. (d) On FSDD.

Figure 4: Impact of classification guidance attenuation rate on AC accuracy.

BPP	Bandwidth reduction	Accuracy	Accuracy loss
1	93.7%	80.93%	1.06%
2	87.4%	81.14%	0.85%
3	81.1%	81.67%	0.32%
4	74.8%	82.05%	-0.06%
5	68.5%	81.92%	0.07%

Table 2: Relation between AE compression ratio and AC accuracy.

rate (*i.e.* $\alpha = 48$ in image datasets, $\alpha = 16$ in audio datasets), we developed multiple CDC models with different compression ratios by adjusting the mask number m of AE.

Table 2 illustrates the AC accuracy of CDC models with different compression ratios as well as the corresponding accuracy loss trained with the preprocessed LSUN dataset (*i.e.* 81.99%)[3]. As we can see, under the experimental setting, compared with training AC with the original dataset (preprocessed LSUN), our approach manages to reduce the bandwidth consumption by 93.7% with an accuracy loss of only 1.06%. According to Table 2, in general, the accuracy loss of CDC increases with the intensity of raw data compression. It should be noted that with a 4 BPP compression ratio, CDC has a higher classification accuracy compared with the baseline AC. We believe that AE with classification guidance from EC on the edge manages to assist the training of AC on the Cloud by successfully preserving classification related features.

To further evaluate CDC's capability on dealing with compression ratio switches (according to the dynamic network bandwidth status in Figure 1), we changed m in different manners and fine-tuned our model with a low learning rate. The variation of MSE under different learning rates and masks are illustrated in Figure 5. As we can see, our model manages to rapidly converge after compression ratio adjustments. It should be noted that when the compression ratio decreases, the training curve does not oscillate and the convergence is quite smooth.

[3]We explicitly present the result of LSUN with images of real-life scenes, where the performance of CDC is relatively representative on such large-scale dataset, for a more convincing discussion.

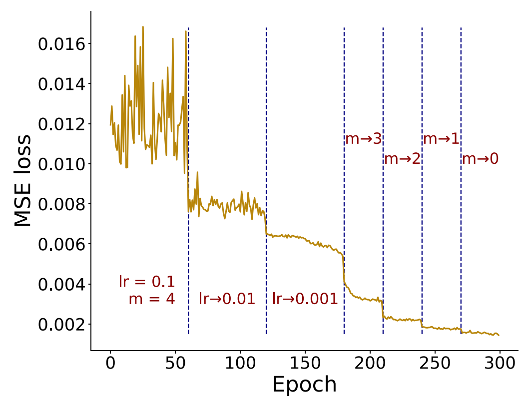


Figure 5: Impact of the compression ratio switching on CDC.

5 Conclusion

In this paper, we present CDC for effective edge-cloud collaborative classifier training, where the tradeoff between network bandwidth consumption and classification accuracy can be achieved. Specifically, we develop a classification-guided autoencoder to compress training data while preserving critical features for the training of high-accuracy classifier on the Cloud, and a quantization based compression method for the bandwidth-accuracy tradeoff. We conducted extensive experiments to evaluate the performance of CDC. For the edge-cloud collaborative training of a DNN classifier (*i.e.* ResNext-29), compared with that based on raw data, CDC manages to consume 14.9× less bandwidth with an accuracy loss no more than 1.06%; compared with DNN training with data compressed by AE without guidance, CDC introduces at least 100% lower accuracy loss.

Acknowledgments

This work was supported by the National Key R&D Program of China (No. 2017YFB1010004), the National Natural Science Foundation of China (Nos. 61772410, 61802298, U1811461, and 11690011), the Fundamental Research Funds for the Central Universities (No. xjj2018237), and the China Postdoctoral Science Foundation (No. 2017M623177).

References

[Abadi *et al.*, 2016] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[Chen and Ran, 2019] Jiasi Chen and Xukan Ran. Deep learning with edge computing: A review. *Proc. IEEE*, 107(8):1655–1674, 2019.

[Cipolla *et al.*, 2018] Roberto Cipolla, Yarin Gal, and Alex Kendall. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proc. of IEEE CVPR*, pages 7482–7491, 2018.

[Cisco, 2016] Cisco. Cisco global cloud index: Forecast and methodology, 2016–2021. *Cisco White paper*, 2016.

[Deng *et al.*, 2009] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proc. IEEE CVPR*, pages 248–255, 2009.

[Hardy *et al.*, 2017] Corentin Hardy, Erwan Le Merrer, and Bruno Sericola. Distributed deep learning on edge-devices: feasibility via adaptive compression. In *Proc. IEEE NCA*, pages 1–8. IEEE, 2017.

[Jackson *et al.*, 2018] Zohar Jackson, Csar Souza, Jason Flaks, and H Nicolas. Jakobovski/free-spoken-digit-dataset v1. 0.7, 2018.

[Kairouz *et al.*, 2019] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

[Le *et al.*, 2018] Lei Le, Andrew Patterson, and Martha White. Supervised autoencoders: Improving generalization performance with unsupervised regularizers. In *Proc. NeurIPS*, pages 107–117, 2018.

[Li *et al.*, 2014] Mu Li, David G Andersen, Jun Woo Park, Alexander J Smola, et al. Scaling distributed machine learning with the parameter server. In *Proc. OSDI*, pages 583–598, 2014.

[Li *et al.*, 2018a] En Li, Zhi Zhou, and Xu Chen. Edge intelligence: On-demand deep learning model co-inference with device-edge synergy. In *Proc. Workshop on Mobile Edge Communications*, pages 31–36. ACM, 2018.

[Li *et al.*, 2018b] He Li, Kaoru Ota, and Mianxiong Dong. Learning iot in edge: deep learning for the internet of things with edge computing. *IEEE Network*, 32(1):96–101, 2018.

[Li *et al.*, 2018c] Mu Li, Wangmeng Zuo, Shuhang Gu, Debin Zhao, and David Zhang. Learning convolutional networks for content-weighted image compression. In *Proc. of IEEE CVPR*, pages 3214–3223, 2018.

[Liu *et al.*, 2016] Tongliang Liu, Dacheng Tao, Mingli Song, and Stephen J Maybank. Algorithm-dependent generalization bounds for multi-task learning. *IEEE PAMI*, 39(2):227–241, 2016.

[Mell *et al.*, 2011] Peter Mell, Tim Grance, et al. The nist definition of cloud computing. 2011.

[Salamon *et al.*, 2014] Justin Salamon, Christopher Jacoby, and Juan Pablo Bello. A dataset and taxonomy for urban sound research. In *Proc. ACM ICME*, pages 1041–1044, 2014.

[Satyanarayan, 2017] Mahadev Satyanarayan. The emergence of edge computing. *Computer*, 50(1):30–39, 2017.

[Shaham and Michaeli, 2018] Tamar Rott Shaham and Tomer Michaeli. Deformation aware image compression. In *Proc. of IEEE CVPR*, 2018.

[Shi and Dustdar, 2016] Weisong Shi and Schahram Dustdar. The promise of edge computing. *Computer*, 49(5):78–81, 2016.

[Smith *et al.*, 2017] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated multi-task learning. In *Proc. of NeurIPS*, pages 4424–4434, 2017.

[Tang *et al.*, 2018] Hanlin Tang, Shaoduo Gan, Ce Zhang, Tong Zhang, and Ji Liu. Communication compression for decentralized training. In *Proc. NeurIPS*, pages 7652–7662, 2018.

[Teerapittayanon *et al.*, 2017] Surat Teerapittayanon, Bradley McDanel, and HT Kung. Distributed deep neural networks over the cloud, the edge and end devices. In *Proc. IEEE ICDCS*, pages 328–339, 2017.

[Theis *et al.*, 2017] Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszár. Lossy image compression with compressive autoencoders. *ICLR*, 2017.

[Xie *et al.*, 2017] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proc. of IEEE CVPR*, pages 1492–1500, 2017.

[Yu *et al.*, 2015] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, et al. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

[Zhou *et al.*, 2018] Lei Zhou, Chunlei Cai, Yue Gao, Sanbao Su, and Junmin Wu. Variational autoencoder for low bit-rate image compression. In *Proc. IEEE CVPR*, pages 2617–2620, 2018.

[Zhou *et al.*, 2019] Zhi Zhou, Xu Chen, En Li, Liekang Zeng, Ke Luo, and Junshan Zhang. Edge intelligence: Paving the last mile of artificial intelligence with edge computing. *Proc. IEEE*, 107(8):1738–1762, 2019.