# Channel-aware Partial Fine-Tuning for AirPooling based Semantic Communications

Nan Wang, Yinglei Teng, *Senior Member, IEEE*, Jiachen Sun,
Beijing Key Laboratory of Space-ground Interconnection and Convergence
Beijing University of Posts and Telecommunications (BUPT), Xitucheng Road No.10, Beijing, China, 100876.
Email: wangnan_26@bupt.edu.cn, lilytengtt@gmail.com, sunjiachen@bupt.edu.cn

***Abstract*—Recently, multiple devices in the Internet of Things collaborate to perceive the environment, offering richer information for intelligent decisions. However, due to the limited network resources, the aggregation of massive sensory data over the dynamic channel environments presents a critical challenge to the real-time learning performance. In this work, we propose a channel-aware partial fine-tuning scheme for the AirPooling based semantic communication system, enabling rapid delivery of accurate models in dynamic channel environments. To achieve efficient information aggregation, Over-the-Air Pooling (AirPooling) is leveraged to facilitate the pooling of multiple single-view features during concurrent transmission, with channel encoding and decoding performed via autoencoder-based neural networks. In addition, to address the performance degradation caused by deteriorating channels, a partial fine-tuning method is proposed, which freezes part of the network layers to reduce the computation burden while maintaining real-time performance. Simulation results show that in dynamic channel environments, the proposed scheme achieves higher learning performance with lower fine-tuning overhead compared to baselines.**

***Index Terms*—over-the-air computation, semantic communication, model fine-tuning, multi-view learning.**

## I. Introduction

THE convergence of Internet of Things and artificial intelligence (AI) has ushered in an era of unprecedented intelligence, where valuable information can be rapidly extracted from massive data for intelligent decision-making [1]. In many AI-empowered emerging applications, many heterogeneous devices collaborate to sense their surroundings and upload the multi-view or multi-modal data to the edge server for comprehensive analysis. However, compared to single-source analysis, multi-source data processing not only needs to address challenges such as data alignment [2] and incompleteness [3] but also imposes greater transmission demand on the network for data aggregation. Therefore, developing efficient data management schemes to enable low-latency collaborative learning has become a primary concern.

To alleviate the impact of the wireless channel on the end-to-end task performance, semantic communications [4], based on joint source-channel coding (JSCC), not only guarantees the semantic integrity of data transmission, but also greatly reduces the transmission overhead. For multi-modal tasks, Zhang *et al*. [5] proposed a unified end-to-end framework, in which the encoded data from different modalities are transmitted through their respective channels to the receiver, followed by aggregation and input to the channel decoder for subsequent processing. For the 3D visual reconstruction task, Fu *et al*. [6] transmitted encoded signals from multiple views to the receiver, where each view's signal is decoded in parallel via deconvolution and then fed into a joint context transfer module to enhance reconstruction quality by exploiting inter-view correlations. Despite coupling learning and communication, the existing "receive-then-aggregate" in collaborative semantic communications may either face degraded decoding performance due to failures in single-source information sampling and transmission or suffer from higher latency caused by the use of separate transmission channels.

In fact, many collaborative tasks focus on the aggregated information rather than the individual source data. In this regard, over-the-air (OTA) computation [7] has emerged as a highly promising communication-efficient solution, leveraging signal superposition to aggregate multiple single-source data during concurrent transmissions. Recent works have been exploring the OTA technique to expedite end-to-end multi-view learning. Liu *et al*. [8] proposed an over-the-air pooling (AirPooling) scheme, which leverages the waveform superposition property of a multiple access channel to perform multi-view pooling during transmission, and optimizes the configuration parameters to balance the aggregation error and the approximation error of Max-AirPooling. Chen *et al*. [9] investigated the impact of channel conditions and view quantity on the inference accuracy of the AirPooling scheme and theoretically demonstrated that the AirPooling scheme achieves superior noise suppression capability compared to the orthogonal multiple access (OMA) scheme. However, all the OTA transceiver designs in the aforementioned works concentrate on the bit error, which is not fully equivalent to task-specific performance such as inference accuracy in learning tasks. The lack of research on the impact of OTA aggregation error on final task performance serves as the motivation for our current work.

In this paper, we propose a channel-aware partial fine-tuning scheme for the AirPooling based semantic communication system to expedite the collaborative learning. First, a collaborative learning framework for multi-view object recognition is constructed, where multiple sensing devices send the joint source-channel coded signals of the single-view features to the edge server for subsequent fusion and recognition. Then, to improve the communication efficiency, the AirPooling-based channel encoder and decoder is designed to enable

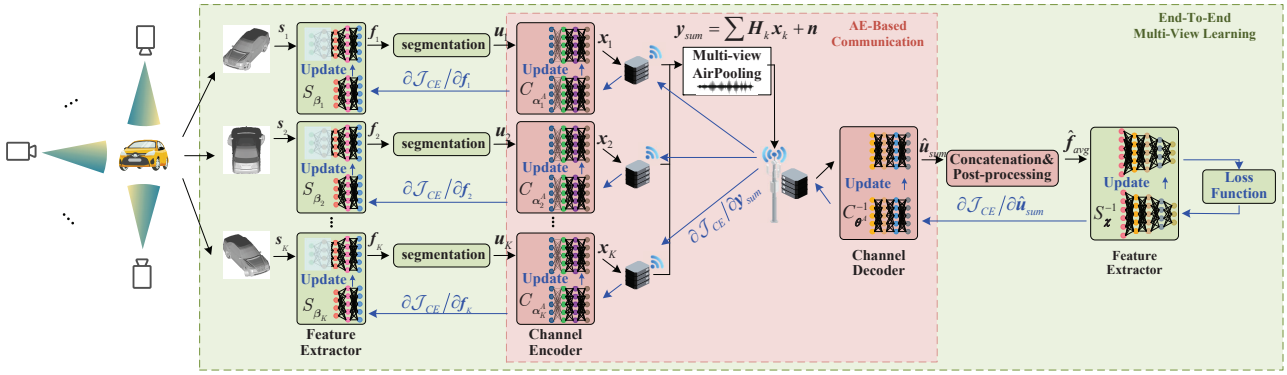



**Fig. 1:** The framework of collaborative semantic communication system.

concurrent transmission of single-view encoded features over a shared channel, allowing the receiver to directly obtain an aggregated feature for subsequent classification. In addition, to mitigate the performance degradation of the pre-trained model under deteriorating channel conditions, we propose a partial fine-tuning approach with layer freezing, effectively enhancing model performance and reducing computation overhead. Simulation results show that in the low signal-to-noise ratio (SNR) regime, the proposed partial fine-tuning scheme reduces computation overhead by 56.1% compared to full fine-tuning scheme, with only a 4.9% accuracy drop.

## II. Collaborative Semantic Communication System

As shown in Fig. 1, we consider a multi-device collaborative learning system for multi-view object recognition, which consists of an edge server equipped with $N_R$ antennas and $K$ sensing devices equipped with $N_T$ antennas. Object recognition is implemented by a multi-view convolutional neural network (MVCNN) [10], which splits the standard convolutional neural network (CNN) into a feature extractor model and a global classifier model, deployed on the sensing devices and edge server, respectively. The feature extractor extracts viewpoint-specific features, while the global classifier makes global predictions based on aggregated features pooled from multiple single views.

### A. Feature Extractor and Channel Encoder

Define the image captured from the $k$-th viewpoint as $\boldsymbol{s}_k$, then the extracted single-view feature map $\boldsymbol{f}_k \in \mathbb{R}^{N_F}$ can be expressed as

$$\boldsymbol{f}_k = S_{\boldsymbol{\beta}_k}(\boldsymbol{s}_k), \tag{1}$$

where $S_{\boldsymbol{\beta}_k}(\cdot)$ is the feature extractor network with the parameter $\boldsymbol{\beta}_k$.

After the feature extraction, the single-view features are encoded into transmit signals through channel encoding. We consider an autoencoder (AE) based communication framework [11], in which neural networks are employed to learn the encoding and decoding of the transmitted message. The single-view feature is divided into $\tau$ vectors, each of which is independently encoded and transmitted to the edge server. The source vector for each transmission is denoted by $\boldsymbol{u}_k \in \mathbb{R}^{N_U}$, i.e., $N_U = N_F/\tau$. Assume a closed-loop MIMO system where the channel state information (CSI) is known to both the sensing devices and edge server. The source vector $\boldsymbol{u}_k$ is concatenated with the channel state and then fed into the channel encoder network $C_{\boldsymbol{\alpha}_k}(\cdot) : \mathbb{R}^{N_U} \times \mathbb{C}^{N_R \times N_T} \to \mathbb{C}^{N_T \times 1}$,

$$\boldsymbol{x}_k = C_{\boldsymbol{\alpha}_k}(\boldsymbol{u}_k, \boldsymbol{H}_k), \tag{2}$$

where $\boldsymbol{\alpha}_k$ is the channel encoder parameter and $\boldsymbol{H}_k \in \mathbb{C}^{N_R \times N_T}$ is the CSI from the sensing device $k$ to the edge server[1]. For the power constraint, a normalization layer is added before the output of the channel encoder to ensure that the transmit power of each sensing device does not exceed $P_T$, i.e., $\mathbb{E}\left[\|\boldsymbol{u}_k\|^2\right] \le P_T$.

To avoid multi-user interference, orthogonal multiple access (OMA) is employed to transmit the encoded signals over their respective channels to the edge server,

$$\boldsymbol{y}_k = \boldsymbol{H}_k \boldsymbol{x}_k + \boldsymbol{n}, \tag{3}$$

where $\boldsymbol{n}$ denotes the additive white Gaussian noise with the distribution $\mathcal{CN}\left(\boldsymbol{0}, \sigma^2 \boldsymbol{I}_{N_R}\right)$.

### B. Channel Decoder and Global Classifier

Similar to the channel encoder, the received signal at the edge server is input to the channel decoder network $C_\theta^{-1}(\cdot) : \mathbb{C}^{N_R \times 1} \times \mathbb{C}^{N_R \times N_T} \to \mathbb{R}^{N_L}$ through the concatenation with CSI,

$$\hat{\boldsymbol{u}}_k = C_{\boldsymbol{\theta}}^{-1}(\boldsymbol{y}_k, \boldsymbol{H}_k), \tag{4}$$

where $\boldsymbol{\theta}$ is the channel decoder parameter. After $\tau$ transmissions, the recovered signals are recombined into single-view features $\hat{\boldsymbol{f}}_k \in \mathbb{R}^{N_F}$, and multiple single-view features are average-pooled into an aggregated feature,

$$\hat{\boldsymbol{f}}_{avg} = \frac{\sum_{k=1}^{K} \hat{\boldsymbol{f}}_k}{K}. \tag{5}$$

[1]The complex-valued CSI is first transformed into a real-valued vector of length $2N_R N_T$, which is then fed into the channel encoder network. The channel encoder outputs a real-valued vector of length $2N_T$, where the first half is treated as the real part and the second half as the imaginary part to form the complex-valued source vector.

Then, the aggregated feature is fed into the global classifier to obtain the predicted results,

$$\hat{\boldsymbol{p}} = S_{\boldsymbol{\chi}}^{-1}\left(\hat{\boldsymbol{f}}_{avg}\right), \tag{6}$$

where $S_{\boldsymbol{\chi}}^{-1}(\cdot)$ is the global classifier network with the parameter $\boldsymbol{\chi}$.

### C. End-to-End Training Procedure

For the multi-view object recognition task, the categorical cross-entropy loss function is adpoted as the training criterion for the end-to-end system, rather than training the learning and communication modules separately. The loss function is defined by

$$\mathcal{J}_{CE}(\boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\theta}, \boldsymbol{\chi}) = -\sum_{z=1}^{Z} \boldsymbol{p}(z) \log\left[\hat{\boldsymbol{p}}(z)\right], \tag{7}$$

where $Z$ denotes the total number of classes in the dataset, and $\boldsymbol{p}(z)$ represents the one-hot encoding of the ground truth labels.

## III. Multi-View AirPooling And Partial Fine-Tuning Scheme

Based on the aggregation requirements of multi-view learning, we utilize an AirPooling aggregation method to further enhance the learning efficiency. Then, considering the performance degradation of the pre-trained model caused by deteriorating channel conditions, a channel-aware partial fine-tuning scheme is designed for the end-to-end learning system.

### A. AirPooling for Multi-View Aggregation

In our multi-view learning system, as the focus is on the pooled feature rather than on each individual one, the "receive-then-aggregate" of single-view features lead to unnecessary resource consumption. Therefore, leveraging the superposition property of the multiple-access channel, we utilize the OTA computation to achieve multi-view pooling during the transmission process. Similar to the OMA mechanism, the scalar $\boldsymbol{u}_k$ is encoded via the AirPooling-based channel encoder network $C_{\boldsymbol{\alpha}_k^{OTA}}(\cdot) : \mathbb{R}^{N_U} \times \mathbb{C}^{N_R \times N_T} \to \mathbb{C}^{N_T \times 1}$,

$$\boldsymbol{x}_k = C_{\boldsymbol{\alpha}_k^{OTA}}(\boldsymbol{u}_k, \boldsymbol{H}_k). \tag{8}$$

Then, multiple sensing devices simultaneously transmit the encoded signals to the edge server,

$$\boldsymbol{y}_{sum} = \sum_{k=1}^{K} \boldsymbol{H}_k \boldsymbol{x}_k + \boldsymbol{n}. \tag{9}$$

Unlike the OMA mechanism, AirPooling does not require to detect individual signals at the edge server. Instead, the aggregated signal is treated as the sum of single-view signals and is directly fed into the AirPooling-based channel decoder $C_{\boldsymbol{\theta}^{OTA}}^{-1}(\cdot) : \mathbb{C}^{N_R \times 1} \times \mathbb{C}^{N_R \times N_T} \to \mathbb{R}^{N_U}$,

$$\hat{\boldsymbol{u}}_{sum} = C_{\boldsymbol{\theta}^{OTA}}^{-1}(\boldsymbol{y}_{sum}, \boldsymbol{H}_{sum}), \tag{10}$$

where $\boldsymbol{H}_{sum} = \sum_{k=1}^{K} \boldsymbol{H}_k$ is the sum of CSI. After recovering the completed feature over $\tau$ transmissions, the edge server only needs to perform a simple post-processing to obtain the aggregated feature,

$$\hat{\boldsymbol{f}}_{avg} = \frac{\hat{\boldsymbol{f}}_{sum}}{K}. \tag{11}$$

Compared to the time-consuming procedure of the OMA scheme, which requires $\mathcal{O}(K)$ transmission resource blocks and $\mathcal{O}(K)$ channel decoding operations, the AirPooling scheme only involves $\mathcal{O}(1)$ transmission resource block and $\mathcal{O}(1)$ channel decoding operation, significantly improving the execution efficiency of the end-to-end model.

### B. Partial Fine-Tuning for Deteriorating Channels

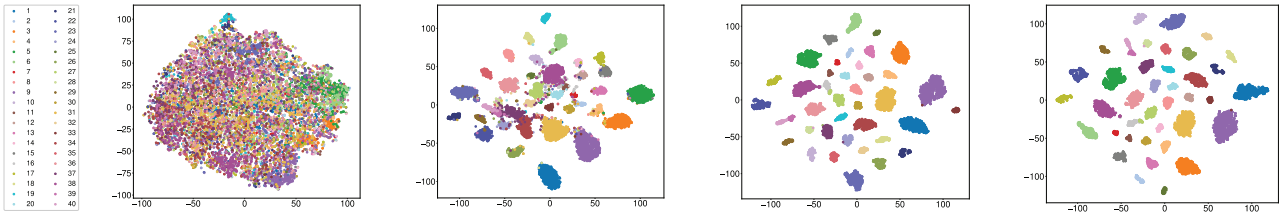
Different from the traditional learning-communication separate design, the proposed semantic communication system for multi-view recognition adopts a joint source–channel encoding approach for processing view information. Therefore, during practical deployment, the test accuracy depends not only on the captured view information but also on the current channel conditions. Let $\boldsymbol{x}_k = \boldsymbol{\phi}_z + \boldsymbol{n}'_k$, where $\boldsymbol{\phi}_z$ is the centroid of class $z$, and $\boldsymbol{n}'_k \sim \mathcal{CN}\left(\boldsymbol{0}, \sigma_v^2 \boldsymbol{I}_{N_T}\right)$ represents the view-specific noise. Then, the received aggregated signal is characterized by the following distribution:

$$\boldsymbol{y}_{sum} \sim \mathcal{CN}\left(\boldsymbol{0}, \left(K\|\boldsymbol{\phi}_z\|^2 + KN_T\sigma_v^2 + \sigma^2\right)\boldsymbol{I}_{N_R}\right). \tag{12}$$

The specific proof is provided in Appendix A. As shown in the above equation, increasing channel noise leads to a larger variance in the aggregated signal, resulting in a more dispersed signal distribution. Such dispersion hinders the ability of the subsequent neural networks to accurately map the aggregated signal to the correct class, ultimately leading to a decline in prediction accuracy. Fig. 2 presents the visualization of final predicted results under different channel conditions during the inference phase. It can be observed that as the channel conditions deteriorate, the predictions become more scattered and disorganized, with the boundaries between classes gradually becoming blurred. Since the neural network is unable to effectively distinguish between different classes, the accuracy decreases accordingly. Therefore, when the system detects that the channel conditions have deteriorated to a certain extent, it is necessary to fine-tune the original pre-trained model to ensure the performance of the real-time model.

However, transmitting all view data to the edge server for model updating raises privacy concerns, and full end-to-end fine-tuning consumes substantial computation resources on the device side. Since channel variations primarily impact the later layers of the model, a partial fine-tuning approach is proposed, utilizing the layer freezing technique to prevent the parameters of earlier layers (such as the feature extractor) from participating in gradient updates, thereby reducing computation overhead. Consider an end-to-end learning model where the first $l$ layers are frozen. The remaining unfrozen trainable layers are updated using a gradient descent algorithm, and the model update process can be formulated as follows,

$$\boldsymbol{w}'_i = \begin{cases} \boldsymbol{w}_i, & i \le l, \\ \boldsymbol{w}_i - \gamma \frac{\partial \mathcal{J}_{CE}}{\partial \boldsymbol{w}_i}, & i > l, \end{cases} \tag{13}$$



(a) SNR = -20dB, ACC = 15.6 %

(b) SNR = -10dB ACC = 81.8%

(c) SNR = 0dB ACC = 94.2%

(d) SNR = 10dB ACC = 94.4%

**Fig. 2:** t-SNE visualizations of predicted results under different channel conditions on ModelNet40 [10], where the model was pretrained with an SNR of 15 dB.

where $\boldsymbol{w}_i$ is denoted as the model parameters of the $i$-th layer, and $\gamma$ is the learning rate.

### C. Implementation Issue of Partial Fine-tuning

As depicted in Fig. 1, by utilizing AirPooling and layer freezing, we propose a partial fine-tuning scheme for the AirPooling-based semantic communication system, which ensures stable high performance of the end-to-end model even under deteriorating channel conditions. Assuming that all the frozen layers are located on the model deployed at the sensing device, the detailed procedure is as follows.

**(1) Device-side Model Forward Propagation:** Using the local feature extractor and channel encoder, the sensing device encodes the image data captured from the specific angle into a single-view feature for transmission.

**(2) Multi-view AirPooling:** With the over-the-air computation technique, all sensors concurrently transmit encoded signals over a shared channel while accomplishing multi-view pooling.

**(3) Server-side Model Forward Propagation:** The edge server processes the received signals through the channel decoder and global classifier to derive the predictions.

**(4) Server-side Model Back Propagation:** Based on the predictions and corresponding labels, the loss is computed and used to update the server-side model parameters via gradient descent.

**(5) Downlink Broadcasting of Received Signal's Gradients:** When back propagation reaches the first layer of the channel decoder, the gradient of the received signal is broadcast to the sensing devices to enable the backward update of the device-side models.

**(6) Device-side Model Back Propagation:** After receiving the gradient of the received signal, each sensing device updates its unfrozen network parameters using the gradient descent algorithm.

Although freezing certain layers may lead to a drop in accuracy, the proposed partial fine-tuning scheme offers advantages in terms of computation and memory overhead. Specially, assume frozen layers include $L_\xi$ convolutional layers and $L_\zeta$ linear layers. Let $C_i^I$, $C_i^O$ represent the input and output channels of the convolutional layers, $G_i$ the kernel size, and $W_i$, $H_i$ the width and height of the output feature map. Denote $N_i^I$ and $N_i^O$ as the number of input and output neurons for each linear layer. Then, the number of operations saved by our partial fine-tuning method can be calculated as $FLOPS = \sum_{i=1}^{L_\xi} 2 \times C_i^I \times C_i^O \times W_i \times H_i \times G_i^2 + \sum_{i=1}^{L_\zeta} 2 \times N_i^I \times N_i^O$. Furthermore, freezing model parameters eliminates the need for devices to allocate memory for storing gradients and intermediate activations in the frozen layers, thereby reducing memory consumption.

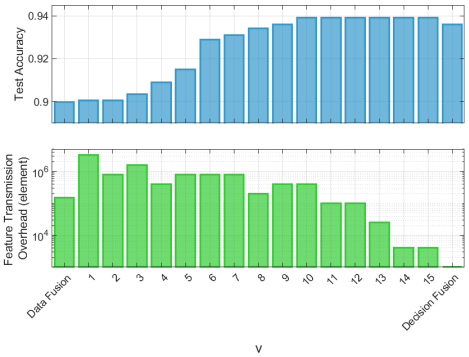



**Fig. 3:** Impact of the view-pooling point on learning performance and transmission overhead.

Another critical factor affecting practical deployment is the view-pooling point $v$, which indicates that the first $v$ layers of the standard CNN are deployed on the sensing devices as the feature extractor, while the remaining layers are deployed on the edge server as the global classifier. Fig. 3 shows the learning performance and the feature transmission overhead by the edge device when selecting different view-pooling points, with VGG16 used as the standard CNN model. It can be observed that deeper view-pooling in VGG16 leads to higher accuracy, as it allows for richer feature extraction and improved cross-view complementarity. Furthermore, in the collaborative learning process, the selection of the view pooling point influences not only the transmission overhead but also the computation overhead. Pooling at deeper layers shifts more computation to sensing devices, which should be carefully considered in both the model fine-tuning and inference phases.

## IV. Simulation and Results

In the simulation, we consider a collaborative semantic communication system consisting of $K = 12$ sensing devices

**TABLE I:** Fine-tuning performance with respect to the number of frozen layer $l$, $v$=10.

| | Frozen Layers | Trained Parameters Size (MB) | Computation Overhead (GFLOPS) | Fine-Tuning Latency(s) | Downloading Overhead (Mbit) | Test Accuracy | |
|---|---|---|---|---|---|---|---|
| | | | | | | $SNR = -20dB$ | $SNR = -15dB$ |
| Full Fine-tuning | $l = 0$ | 134.79 | 49.93 | 226.00 | 3.21 | 0.906 | 0.932 |
| Device-side Model Fine-tuning | $l = 2$ | 134.75 | 46.02 | 216.52 | 3.21 | 0.890 | 0.929 |
| | $l = 4$ | 134.53 | 40.46 | 210.80 | | 0.886 | 0.927 |
| | $l = 6$ | 133.64 | 38.61 | 205.47 | | 0.882 | 0.926 |
| | $l = 8$ | 131.87 | 29.34 | 197.47 | | 0.871 | 0.922 |
| | $l = 10$ | 127.15 | 21.94 | 194.12 | | 0.857 | 0.921 |
| | $l = 12$ | 126.98 | 20.87 | 174.97 | | 0.845 | 0.920 |
| Device-side Model Frozen | $l = 14$ | 126.97 | 20.80 | 165.53 | 0 | 0.842 | 0.917 |
| | $l = 16$ | 126.80 | 19.72 | 147.42 | | 0.839 | 0.917 |
| | $l = 18$ | 126.79 | 19.66 | 141.05 | | 0.828 | 0.916 |
| | $l = 20$ | 122.07 | 17.81 | 138.78 | | 0.793 | 0.910 |
| | $l = 22$ | 16.95 | 16.68 | 102.40 | | 0.647 | 0.882 |

and an edge server, with the antenna size $N_R = N_T = 16$. To implement multi-view learning, VGG16 is adopted as the standard CNN backbone, with the view-pooling point placed after the tenth convolutional layer, i.e., $v$ = 10. We consider ModelNet40 dataset for multi-view object recognition, where each of the 12 views per object is captured by a sensor. Both the channel encoder and decoder networks consist of four fully connected layers. Each MIMO channel is assumed to be Rayleigh fading, and the hardware noise factor is set as 6 dB. The pretrained model is trained with SNR = $KP_T/\sigma^2$ = 5 dB and a learning rate of 5e-5. Fine-tuning is performed for 30 epochs with a learning rate set to 5e-6. All simulations were performed using a single NVIDIA RTX 3090 GPU.

For performance comparison, we considered the OMA-based scheme and separate source-channel coding (SSCC) with AirPooling scheme. In the OMA-based scheme, the total system bandwidth is equally allocated among the sensing devices, which transmit encoded signals through their respective channels. The edge server performs channel decoding in sequence, and then conducts view pooling. And in the SSCC with AirPooling scheme, the beamforming design is derived using differential geometry to minimize the mean square error [12].

As shown in Table I, we compare the performance of different fine-tuning schemes in terms of test accuracy and resource overhead. As the number of frozen layers increases, the model's learning performance slightly declines, while the fine-tuning latency is correspondingly reduced. Compared to the full fine-tuning scheme, the partial fine-tuning scheme with $l$ = 10 reduces the computation overhead by 56.1%. Even under extremely harsh channel conditions with SNR = –20 dB, this scheme achieves a test accuracy only 4.9% lower than that of the full fine-tuning. Furthermore, when SNR = –15 dB, the test accuracy remains above 90% even with up to $l$ = 20, at which point the entire model on the sensing device is frozen, and the downloading of aggregated signal's gradient can also be omitted.

In Fig. 4, we further investigate the learning performance versus the number of fine-tuning sensing devices under different fine-tuning schemes. As the number of fine-tuning

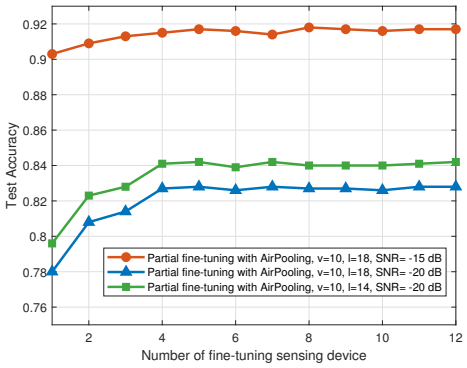



**Fig. 4:** Test accuracy versus the number of fine-tuning sensing device.

sensing devices increases, the test accuracy of the fine-tuned model gradually improves. This is because the model can better learn to extract meaningful semantic information from more view features affected by the current channel conditions. However, when the number of fine-tuning sensing devices reaches 4, the performance improvement tends to saturate, and adding more devices yields negligible gains in accuracy. Moreover, under more severe channel conditions, involving more devices in fine-tuning has a more pronounced impact on model performance. Therefore, in practical deployments, the selection of fine-tuning devices should take into account their energy efficiency to achieve a balance between learning performance and resource consumption.

Next, we compare the learning performance under different schemes in Fig. 5. It can be observed that the AirPooling-based scheme demonstrates stronger robustness to noise compared to the OMA-based scheme, maintaining high accuracy even when SNR drops to –5 dB. In contrast, the no fine-tuning with OMA scheme suffers a significant degradation in accuracy when SNR falls below 0 dB. This is because, in the OMA-based scheme, each single-view feature is individually affected by channel noise during transmission and then sequentially processed by channel decoders to recover the original signal. Such multiple decoding stages can accumulate errors, leading

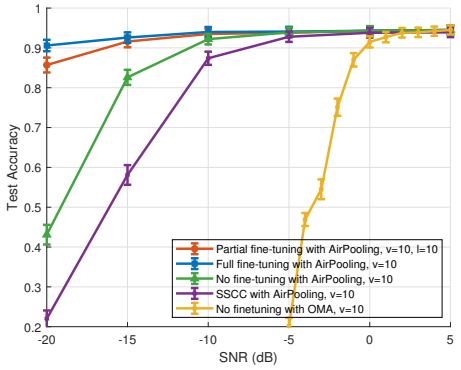



**Fig. 5:** Test accuracy versus SNR.

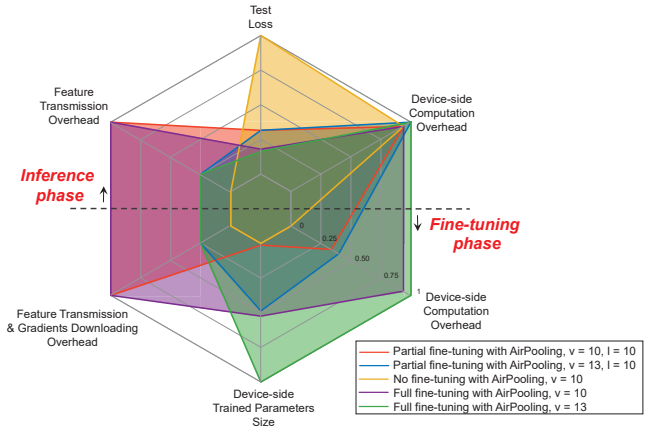



**Fig. 6:** System performance comparison with $SNR = -20dB$. All the values are normalized.

to increased deviation. In the low SNR regime, the proposed partial fine-tuning scheme significantly outperforms both the no fine-tuning scheme and the SSCC with AirPooling scheme. This improvement stems from the proposed scheme's ability to perceive both data semantics and channel noise, enabling it to adjust network parameters according to the current channel conditions for optimal learning performance. In contrast, the no fine-tuning scheme lacks adaptability to worsening channel conditions, resulting in a sharp decline in test accuracy as the channel quality further deteriorates.

Fig. 6 shows the learning performance and resource consumption under different schemes. It can be observed that, compared with the no fine-tuning and full fine-tuning schemes, the proposed partial finetuning scheme achieves lower test loss with a moderate increase in finetuning overhead. In addition, we investigate the impact of the view-pooling point on the device-side overhead. When the later layer is selected as the view-pooling point, the computation overhead on the device increases, while the transmission overhead decreases. Therefore, in practical deployments, the selection of the view-pooling point should consider both the available computation resources on the sensing devices and the system's transmission capacity.

## V. Conclusion

In this work, we propose a channel-aware partial fine-tuning scheme for the AirPooling based semantic communication system to address the issues of aggregation latency and semantic distortion. With the AirPooling mechanism, the single-view encoded symbols from multiple sensors are concurrently transmitted to the edge server over a shared channel. And to enhance the model's robustness under varying channel conditions, a partial fine-tuning scheme is designed, which updates only the later layers closer to the wireless channel, thereby reducing the fine-tuning overhead. Experiments show that the proposed partial fine-tuning scheme can ensure better model performance in the dynamic channel environment with lower transmission and computation overhead.

## Appendix A
## Proof of Distribution of the Received Aggregated Signal

The received aggregated signal can be formulated as

$$\begin{aligned}\boldsymbol{y}_{sum} &= \sum_{k=1}^{K} \boldsymbol{H}_k \boldsymbol{x}_k + \boldsymbol{n} \\ &= \sum_{k=1}^{K} \boldsymbol{H}_k \left(\boldsymbol{\phi}_z + \boldsymbol{n}'_k\right) + \boldsymbol{n} \\ &= \left(\sum_{k=1}^{K} \boldsymbol{H}_k \boldsymbol{\phi}_z\right) + \left(\sum_{k=1}^{K} \boldsymbol{H}_k \boldsymbol{n}'_k + \boldsymbol{n}\right) \\ &= \boldsymbol{y}_s + \boldsymbol{n}_{total},\end{aligned} \tag{14}$$

where $\boldsymbol{y}_s = \sum_{k=1}^{K} \boldsymbol{H}_k \boldsymbol{\phi}_z$ denotes the ground-truth signal and $\boldsymbol{n}_{total} = \sum_{k=1}^{K} \boldsymbol{H}_k \boldsymbol{n}'_k + \boldsymbol{n}$ denotes the total noise. Assume each MIMO channel is Rayleigh fading, i.e., $[\boldsymbol{H}_k]_{\iota,o} \sim \mathcal{CN}(0,1), \iota \in \{1,2,\ldots,N_R\}, o \in \{1,2,\ldots,N_T\}$, then we have $[\boldsymbol{H}_{sum}]_{\iota,o} = \sum_{k=1}^{K} [\boldsymbol{H}_k]_{l,o} \sim \mathcal{CN}(0,K)$

The mean, variance, and covariance of the ground-truth signal can be calculated as follows:

$$\mathbb{E}\left[[\boldsymbol{y}_s]_\iota\right] = \sum_{o=1}^{N_T} \mathbb{E}\left[[\boldsymbol{H}_{sum}]_{\iota,o}\right] \phi_{z,o} = 0. \tag{15}$$

$$\begin{aligned}\mathbb{E}\left[\left|[\boldsymbol{y}_s]_\iota\right|^2\right] &= \mathbb{E}\left[\left|\sum_{o=1}^{N_T} [\boldsymbol{H}_{sum}]_{\iota,o} \phi_{z,o}\right|^2\right] \\ &= \sum_{o=1}^{N_T} \mathbb{E}\left[\left|[\boldsymbol{H}_{sum}]_{\iota,o}\right|^2\right] |\phi_{z,o}|^2 \\ &= K \sum_{o=1}^{N_T} |\phi_{z,o}|^2 = K \left\|\boldsymbol{\phi}_z\right\|^2.\end{aligned} \tag{16}$$

$$\mathbb{E}\left[[\boldsymbol{y}_s]_\iota [\boldsymbol{y}_s]^*_{\iota'}\right] = \sum_{\iota,\iota'} \mathbb{E}\left[[\boldsymbol{H}]_{\iota,o} [\boldsymbol{H}]^*_{\iota',o'}\right] \phi_{z,o} \phi^*_{z,o'} = 0, \quad \iota \neq \iota'. \tag{17}$$

Therefore, the ground-truth signal follows the distribution $\boldsymbol{y}_s \sim \mathcal{CN}\left(\boldsymbol{0}, K\left\|\boldsymbol{\phi}_z\right\|^2 \boldsymbol{I}_{N_R}\right)$.

The mean, variance, and covariance of the viewpoint noise $\boldsymbol{n}_k^{view} = \boldsymbol{H}_k \boldsymbol{n}_k'$ can be calculated as follows:

$$\mathbb{E}\left[\left[\boldsymbol{n}_k^{view}\right]_\iota\right] = \sum_{o=1}^{N_T} \mathbb{E}\left[\left[\boldsymbol{H}_k\right]_{\iota,o}\right] \mathbb{E}\left[\left[\boldsymbol{n}_k'\right]_o\right] = 0 \tag{18}$$

$$\mathbb{E}\left[\left|\left[\boldsymbol{n}_k^{view}\right]_\iota\right|^2\right] = \sum_{o=1}^{N_T} \mathbb{E}\left[\left|\left[\boldsymbol{H}_k\right]_{\iota,o}\right|^2\right] \mathbb{E}\left[\left|\left[\boldsymbol{n}_k'\right]_o\right|^2\right] = N_T \sigma_v^2 \tag{19}$$

$$\begin{aligned} \mathrm{Cov}\left(\boldsymbol{n}_k^{view}\right) &= \mathbb{E}\left[\left(\boldsymbol{H}_k \boldsymbol{n}_k'\right)\left(\boldsymbol{H}_k \boldsymbol{n}_k'\right)^H\right] \\ &= \mathbb{E}\left[\boldsymbol{H}_k \boldsymbol{n}_k' \boldsymbol{n}_k'^H \boldsymbol{H}_k^H\right] \\ &= \mathbb{E}\left[\boldsymbol{H}_k \left(\boldsymbol{n}_k' \boldsymbol{n}_k'^H\right) \boldsymbol{H}_k^H\right] \\ &= \mathbb{E}\left[\boldsymbol{H}_k \left(\sigma_v^2 \boldsymbol{I}_{N_T}\right) \boldsymbol{H}_k^H\right] \\ &= \sigma_v^2 \mathbb{E}\left[\boldsymbol{H}_k \boldsymbol{H}_k^H\right] \\ &= \sigma_v^2 \sum_{o=1}^{N_T} \mathbb{E}\left[\boldsymbol{H}_{k,o} \boldsymbol{H}_{k,o}^H\right] \\ &= N_T \sigma_v^2 \boldsymbol{I}_{N_R}. \end{aligned} \tag{20}$$

Therefore, the viewpoint noise follows the distribution $\boldsymbol{n}_k^{view} \sim \mathcal{CN}\left(\boldsymbol{0}, \sigma_v^2 N_T \boldsymbol{I}_{N_R}\right)$. Then, we have $\sum_{k=1}^K \boldsymbol{H}_k \boldsymbol{n}_k' = \sum_{k=1}^K \boldsymbol{n}_k^{view} \sim \mathcal{CN}\left(\boldsymbol{0}, K N_T \sigma_v^2 \boldsymbol{I}_{N_R}\right)$. Since $\boldsymbol{n} \sim \mathcal{CN}\left(\boldsymbol{0}, \sigma^2 \boldsymbol{I}_{N_R}\right)$, $\boldsymbol{n}_{total}$ follows the distribution $\mathcal{CN}\left(\boldsymbol{0}, \left(K N_T \sigma_v^2 + \sigma^2\right) \boldsymbol{I}_{N_R}\right)$.

Therefore, the received aggregated signal follows the distribution $\mathcal{CN}\left(\boldsymbol{0}, \left(K \left\|\boldsymbol{\phi}_z\right\|^2 + K N_T \sigma_v^2 + \sigma^2\right) \boldsymbol{I}_{N_R}\right)$.

## ACKNOWLEDGMENT

This work was supported in part by the National Key R&D Program of China (No. 2021YFB3300100), and the National Natural Science Foundation of China (No. 62171062) and (U24A20234).

## REFERENCES

[1] X. Yan, S. Hu, Y. Mao, Y. Ye, and H. Yu, "Deep multi-view learning methods: A review," *Neurocomputing*, vol. 448, pp. 106–129, 2021.

[2] Z. Chen, J. Chen, W. Zhang, L. Guo, Y. Fang, Y. Huang, Y. Zhang, Y. Geng, J. Z. Pan, W. Song *et al.*, "Meaformer: Multi-modal entity alignment transformer for meta modality hybrid," in *Proceedings of the 31st ACM International Conference on Multimedia*, 2023, pp. 3317–3327.

[3] G. A. Khan, J. Khan, T. Anwar, Z. Ashraf, M. H. Javed, and B. Diallo, "Weighted concept factorization based incomplete multi-view clustering," *IEEE Trans. Artif. Intell.*, 2024.

[4] W. Yang, H. Du, Z. Q. Liew, W. Y. B. Lim, Z. Xiong, D. Niyato, X. Chi, X. Shen, and C. Miao, "Semantic communications for future internet: Fundamentals, applications, and challenges," *IEEE Commun. Surveys Tuts.*, vol. 25, no. 1, pp. 213–250, 2022.

[5] G. Zhang, Q. Hu, Z. Qin, Y. Cai, G. Yu, and X. Tao, "A unified multitask semantic communication system for multimodal data," *IEEE Trans. Commun.*, 2024.

[6] Z. Z. J. Y. Fu, Yuechun and T. Q. Quek, "A semantic communication scheme for distributed holographic-type communications with multi-view images," *IEEE Commun. Lett.*, 2024.

[7] Y. Sun, S. Zhou, Z. Niu, and D. Gündüz, "Dynamic scheduling for over-the-air federated edge learning with energy constraints," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 1, pp. 227–242, 2021.

[8] Z. Liu, Q. Lan, A. E. Kalør, P. Popovski, and K. Huang, "Over-the-air multi-view pooling for distributed sensing," *IEEE Trans. Wirel. Commun.*, 2023.

[9] X. Chen, K. B. Letaief, and K. Huang, "On the view-and-channel aggregation gain in integrated sensing and edge ai," *IEEE J. Sel. Areas Commun.*, 2024.

[10] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, "Multi-view convolutional neural networks for 3d shape recognition," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 945–953.

[11] J. Song, C. Häger, J. Schröder, T. J. O'Shea, E. Agrell, and H. Wymeersch, "Benchmarking and interpreting end-to-end learning of mimo and multi-user communication," *IEEE Trans. Wirel. Commun.*, vol. 21, no. 9, pp. 7287–7298, 2022.

[12] G. Zhu and K. Huang, "Mimo over-the-air computation for high-mobility multimodal sensing," *IEEE Internet Things J.*, vol. 6, no. 4, pp. 6089–6103, 2018.