O2SC: Realizing Channel-Adaptive Semantic Communication with One-Shot Online-Learning

Guangyi Zhang, Kai Kang, Yunlong Cai, Qiyu Hu, Yonina C. Eldar, and A. Lee Swindlehurst

***Abstract*—Motivated by progress in data-driven supervised learning, semantic communication has witnessed remarkable advancements in improving the efficiency of data transmission under various channel conditions. These advancements typically require a substantial amount of training data for offline training, which is challenging in practical systems. Therefore, in this work, we propose O2SC, a one-shot online-learning framework for semantic communication to achieve adaptive transmission under different channel conditions. Since semantic communication relies on acquired channel state information (CSI), we jointly design the channel estimation and semantic communication processes. Specifically, we introduce a denoising module based on one-shot self-supervised learning, allowing semantic communication systems to adapt to new channel conditions without the need to collect extensive training data. The denoising module is utilized to eliminate noise in the received data samples, using only the data samples themselves. Following this, we further exploit meta-learning to allow the system to quickly adapt to diverse channel conditions, by finding an appropriate initialization for each data sample in a timely way. Simulation results demonstrate that the proposed method achieves performance close to that of supervised learning-based approaches while also providing improved generalizability across different channel conditions.**

***Index Terms*—Channel estimation, deep learning, one-shot learning, semantic communication.**

I. Introduction

Building on Shannon's theory [1], wireless communication has evolved from its analog origins in 1G systems to current 5G systems and beyond [2]. Driven by innovative applications and intelligent services such as extended reality and intelligent transportation [3], the landscape of wireless communication is continuously evolving. To meet the demands of these emerging technologies, there has been a surge in semantic communication research in both academia and industry. The growing trend of semantic communication aims at accurately recovering the statistical structure of the underlying information of the source signal and designing the transceiver in an end-to-end fashion [4]–[10].

Semantic communication leverages the advanced deep learning mechanisms, specifically autoencoder architectures [11]–[14]. The key idea of semantic communication is exploiting deep neural networks (DNNs) to extract valuable semantic information in the source data for specific transmission tasks. As a result, semantic communication is able to benefit from high compression ratios and transmission efficiency. This approach is often more effective than the conventional paradigm of separate source and channel coding, as exemplified by techniques such as combining Better Portable Graphics (BPG) source compression with low-density parity-check (LDPC) channel coding [15]. In particular, a landmark semantic communication system for image transmission was developed in [16], where the authors conceived of using DNNs to directly map source data to channel symbols in a joint source-channel-coding (JSCC) manner.

The deep JSCC (DJSCC) model proposed in [16] is trained under specific channel conditions, such as a specific signal-to-noise ratio (SNR), particularly for image transmission tasks. As a result, its performance surpasses that of separated coding scheme designs when dealing with different channel environments. However, DJSCC is most effective within a fixed SNR range and needs to be retrained when the SNR changes. To overcome this problem, the authors of [17] proposed a channel-adaptive semantic communication system to support a range of SNR values, with the aid of an attention mechanism. Recent advancements have been made in [18] and [19] that allow for the transmission rate to be adjusted within a dynamic model, adapting to various channel conditions.

Although previous research has demonstrated promising performance, many of these studies primarily focus on single-input single-output (SISO) scenarios over an additive white Gaussian noise (AWGN) channel. This simplistic model does not accurately capture the complexities of real-world communication channels. To address this limitation and better adapt to the intricacies of practical channel conditions, researchers have explored a series of physical layer design approaches for semantic communication [20]–[25]. In [21], the authors integrated orthogonal frequency division multiplexing (OFDM) with semantic communication to enable wireless image transmission in the presence of multi-path fading. In addition, a vision Transformer-based semantic communication system has been proposed to adaptively learn semantic feature mappings and power allocation strategies according to channel conditions in multiple-input multiple-output (MIMO) systems [22]. In [23], the authors designed a spatial multiplexing mechanism to realize adaptive coding rate allocation by jointly considering the entropy distribution of the source semantic features and wireless MIMO channels. In [24], the authors aimed to adaptively adjust the channel feedback overhead for images with different predicted reconstruction qualities,

Parts of this paper were presented at the 2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring), Florence, Italy.

This work was supported in part by the

G. Zhang, K. Kang, Y. Cai, and Q. Hu are with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou 310027, China (e-mail: zhangguangyi@zju.edu.cn; kangkai@zju.edu.cn; ylcai@zju.edu.cn; qiyhu@zju.edu.cn).

Yonina C. Eldar is with the Department of Mathematics and Computer Science, Weizmann Institute of Science, Rehovot 7610001, Israel (e-mail: yonina.eldar@weizmann.ac.il).

A. Lee Swindlehurst is with the Center for Pervasive Communications and Computing, University of California at Irvine, Irvine, CA 92697 USA (e-mail: swindle@uci.edu).

thereby enhancing transmission reliability.

Despite the enormous potential of these methods in realizing adaptive transmission over various channel conditions, they still face some practical challenges:

- *Data Collection:* The aforementioned deep learning-based semantic communication methods are typically based on supervised learning, which rely on a substantial number of samples to optimize the network parameters during offline training. However, in practice, obtaining a sufficient amount of data samples related, for example, to channel state information (CSI), can be infeasible. This makes it difficult to develop fully adaptive systems that rely solely on offline training.
- *Environmental Variability:* Communication systems must grapple with the complexity and variability of channel environments, which are often unpredictable. Additionally, the distribution of the test data may significantly differ from that of the training data. Many existing studies assume that if a semantic system performs well in both training and validation environments, it will likely generalize effectively to new, unseen environments. However, this assumption is often hard to realize, due to unsatisfactory generalizability of DNN models.

To address the first problem, we attempt to introduce self-supervised learning to further capture the data characteristics, so as to improve the system performance by making full use of channel and source data [26], [27], which has the potential to reduce the need for training data. For the second issue, we conceive of using an online manner by online updating the DNN models to adapt to the current environment. Moreover, to support low-latency real-time data adaptation, the online training process should be accelerated as much as possible. One promising avenue is meta-learning, which aims to improve the learning ability of neural networks by learning from data with different distributions [28]–[32]. Notably, in [33] and [34], the model-agnostic meta-learning (MAML) algorithm and the Reptile algorithm have been introduced. These techniques focus on learning an effective model initialization that enables rapid adaptation to new data samples.

Therefore, inspired by these insights, we seek to solve the above problems by introducing a one-shot online-learning framework. Our primary objective is to establish an adaptive semantic communication system suitable for practical applications. The proposed one-shot online-learning framework is based on exploiting one-shot self-supervised learning in the deep learning-based semantic communication model of [16]. The main idea is to denoise the noisy estimated signal online using the noisy estimated signal itself as the sole training data, which is why we refer to it as the *one-shot online-learning* method. Moreover, previous works typically assume perfect knowledge of the CSI at the transceiver, which is challenging to achieve in real-world communication systems. In contrast, we consider a more practical scenario where the estimation error is taken into consideration. We specifically takes into account the critical aspect of channel data acquisition, by jointly designing the channel estimation and semantic transmission in an end-to-end manner. Furthermore, we devise a meta-learning-based method to accelerate the convergence of the online one-shot training procedure, so that the system can quickly adapt to changing channel conditions. We summarize our main contributions as follows.

(1) We propose O2SC, a one-shot online-learning framework for semantic communication based on one-shot self-supervised learning. It is distinct from existing methods [16]–[24] that utilize a substantial number of source and channel samples to design the semantic communication systems. Our proposed framework operates using an online update scheme, allowing it to utilize real-time data as training samples without the need for accumulating extensive training data.

(2) The proposed O2SC comprises two crucial components: a semantic transmission module (STM) and a channel estimation module (CEM). We view the obtained results from STM and CEM as a noisy degradation of the original sample, and aim to recover the original sample through solving an denoising problem. In particular, we devise two plug-in denoising modules based on self-supervised learning, which are inserted at the end of the STM and CEM modules. For channel estimation, we obtain a preliminary estimate based on a traditional algorithm, and then use the denoising module to obtain a more accurate estimate. For semantic transmission, the denoising module takes the output of the decoder as input and returns a more accurate received signal. The parameters of the denoising module are periodically adjusted for individual samples, ensuring stronger generalizability since it does not rely on training on large data sets from a specific distribution.

(3) Given that the denoising module undergoes online updates when a new real-time data sample becomes available, we propose a meta-learning-based initialization method for O2SC. This method equips the system with excellent generalizability and enables rapid adaptation to changing channel conditions. The proposed method comprises two key components: the inner update and the outer update. These components involve training on samples derived from different data distributions, allowing us to determine a good model initialization. We further propose an online meta-adaptation strategy designed to continually find a good initialization for each future sample while fitting the current data sample, which significantly improves the O2SC efficiency.

We demonstrate the effectiveness of the proposed framework on image sources and practical massive MIMO channels. Simulation results show that our proposed approach achieves performance comparable to traditional supervised learning-based methods and has better generalizability.

The rest of this paper is structured as follows. Section II introduces the proposed one-shot online-learning framework. Section III presents the detailed architecture of the plug-in denoising and Section IV introduces the meta-based fast adaptation strategy. Simulation results are presented in Section V, and finally Section VI concludes the paper.

Notation: Scalars, vectors and matrices are respectively denoted by lower case, boldface lower case and boldface upper case letters. The matrix $\boldsymbol{I}$ represents an identity matrix and $\mathbf{0}$ denotes an all-zero matrix. For a matrix $\boldsymbol{A}$, $\boldsymbol{A}^T$, $\boldsymbol{A}^*$, $\boldsymbol{A}^H$, and $\|\boldsymbol{A}\|_F$ denote its transpose, conjugate, conjugate transpose and

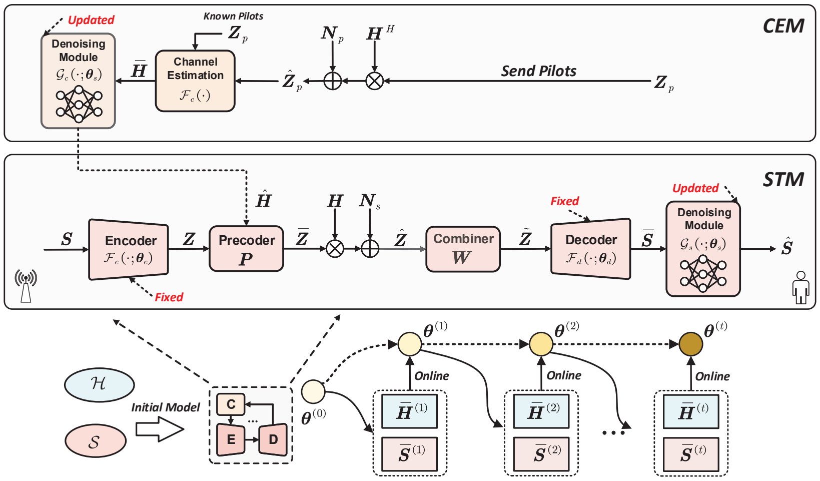


Fig. 1. Illustration of the proposed one-shot online-learning framework.

Frobenius norm, respectively. For a square matrix $\boldsymbol{A}$, Tr$(\boldsymbol{A})$ the trace. In addition, $\mathbb{C}^{m\times n}$ $(\mathbb{R}^{m\times n})$ are the space of $m \times n$ complex (real) matrices.

II. Problem Formulation and One-Shot Framework

In this section, we introduce the STM and CEM and describe the overall O2SC procedure. We then present the proposed one-shot online-learning framework.

A. Problem Formulation

As shown in Fig. 1, we consider a downlink wireless image transmission system, which aims to transmit the image $\boldsymbol{S} \in \mathbb{R}^{C\times H\times W}$ from the transmitter to the receiver, where C, H, and W indicate the dimensions of $\boldsymbol{S}$. Before transmission, $\boldsymbol{S}$ undergoes processing to yield the corresponding encoded feature matrix $\boldsymbol{Z} \in \mathbb{C}^{d\times N_s}$ using an encoder denoted as $\mathcal{F}_e(\cdot)$. Here, N_s denotes the number of channel uses and d denotes the number of data streams for each channel use. Thus, we have $\boldsymbol{Z} = \mathcal{F}_e(\boldsymbol{S};\boldsymbol{\phi}_e)$, where $\boldsymbol{\phi}_e$ represents the trainable parameters of the encoder. Additionally, we define the channel bandwidth ratio as $\rho \triangleq N_s/(C\times H\times W)$.

For the transmission of $\boldsymbol{Z}$, CSI-based MIMO techniques are introduced to further improve the efficiency, as shown in Fig. 1. Specifically, we assume that the transmitter is equipped with N_t transmit antennas and the receiver is equipped with N_r receive antennas. To obtain the CSI matrix $\boldsymbol{H} \in \mathbb{C}^{N_r\times N_t}$, we employ channel estimation for semantic communication to study the impact of estimation errors. Our approach employs a MIMO system in a time-division duplex (TDD) mode in which the downlink CSI can be estimated by exploiting the channel reciprocity. In particular, the receiver sends pilots $\boldsymbol{Z}_p \in \mathbb{C}^{N_r\times L}$ to the transmitter, where L is the length of the pilot. The received pilots $\hat{\boldsymbol{Z}}_p \in \mathbb{C}^{N_t\times L}$ at the transmitter is given by $\hat{\boldsymbol{Z}}_p = \boldsymbol{H}^H\boldsymbol{Z}_p + \boldsymbol{N}_p$, where $\boldsymbol{N}_p \in \mathbb{C}^{N_t\times L}$ denotes AWGN. With $\boldsymbol{Z}_p$ and $\hat{\boldsymbol{Z}}_p$, the estimated downlink CSI matrix $\bar{\boldsymbol{H}}$ is acquired by an estimator $\mathcal{F}_c(\cdot)$ at the transmitter, which is denoted by $\bar{\boldsymbol{H}} = \mathcal{F}_c(\hat{\boldsymbol{Z}}_p, \boldsymbol{Z}_p)$. Based on $\bar{\boldsymbol{H}}$, we exploit singular-value decomposition (SVD) for semantic transmission, i.e., $\bar{\boldsymbol{H}} = \bar{\boldsymbol{P}}\bar{\boldsymbol{\Sigma}}\bar{\boldsymbol{W}}$, where $\bar{\boldsymbol{\Sigma}}$ denotes the diagonal matrix. Since the number of data streams for each channel use is d, the exact precoding matrix is given by $\boldsymbol{P} \in \mathbb{C}^{N_t\times d}$, which is obtained by drawing the corresponding columns from $\bar{\boldsymbol{P}}$. Moreover, at the receiver, we consider zero-forcing detection, the combing matrix is given by $\bar{\boldsymbol{W}} = (\boldsymbol{H}\boldsymbol{P})^{\dagger}$. As a result, the precoded feature matrix is $\bar{\boldsymbol{Z}} = \boldsymbol{P}\boldsymbol{Z}$. The transmitted signal is still subject to a power constraint P_t, i.e., $\|\boldsymbol{P}\boldsymbol{Z}\|_F^2 \le N_s P_t$, where we set $P_t = 1$ without loss of generality. Moreover, we define the average SNR at the receiver as SNR $\triangleq 10\log_{10}\frac{P_t}{\sigma^2}$, where σ^2 denotes the noise variance.

The received signal at the receiver is given by

$$\hat{\boldsymbol{Z}} = \boldsymbol{H}\bar{\boldsymbol{Z}} + \boldsymbol{N}, \tag{1}$$

where $\boldsymbol{N} \sim \mathcal{CN}(\boldsymbol{0}, \sigma^2\boldsymbol{I})$ denotes AWGN. Subsequently, the detected signal after combining is given by

$$\tilde{\boldsymbol{Z}} = \boldsymbol{W}\hat{\boldsymbol{Z}}. \tag{2}$$

Finally, $\tilde{\boldsymbol{Z}}$ is fed into the decoder $\mathcal{F}_d(\cdot)$ to recover the transmitted image, i.e., $\bar{\boldsymbol{S}} = \mathcal{F}_d(\tilde{\boldsymbol{Z}};\boldsymbol{\phi}_d)$, where $\boldsymbol{\phi}_d$ denotes its trainable parameters. The target of the decoder is to estimate $\bar{\boldsymbol{S}}$ close to $\boldsymbol{S}$. Thus, the parameters of the encoder and decoder are updated by solving the following optimization problem:

$$(\boldsymbol{\phi}_e^*, \boldsymbol{\phi}_d^*) = \arg\min_{\boldsymbol{\phi}_e,\boldsymbol{\phi}_d} \mathbb{E}_{p(\boldsymbol{S},\bar{\boldsymbol{S}})}[d(\boldsymbol{S},\bar{\boldsymbol{S}})], \tag{3}$$

where $d(\cdot)$ denotes the distortion function. The distortion function is also used as a measurement of image reconstruction quality, which is defined as $d(\boldsymbol{S},\bar{\boldsymbol{S}}) = \|\boldsymbol{S}-\bar{\boldsymbol{S}}\|_F^2$.

B. One-Shot Online-Learning Framework

We refer to the mapping from $\boldsymbol{S}$ to $\bar{\boldsymbol{S}}$ as the STM and similarly define the mapping from $\boldsymbol{Z}_p$ to $\bar{\boldsymbol{H}}$ as the CEM. In existing works, the STM and CEM are optimized separately to achieve their own targets, i.e., the CEM is optimized to minimize the CSI estimation error, while the STM is optimized to find a parameter set that minimizes the distortion function under different channel conditions. In prior work, they are both optimized over a huge number of data samples, e.g., CSI matrices and images. Although the model can be trained to achieve high performance under the given training conditions, the channel conditions are variable, and there is always performance degradation when the testing conditions differ from those during training. To address this issue, we propose a one-shot online-learning framework by designing two plug-in denoising modules, $\mathcal{G}_c(\cdot)$ and $\mathcal{G}_s(\cdot)$, as illustrated in Fig. 1.

In particular, our target is to quickly adapt STM and CEM to new channel distributions without collecting numerous data samples. Instead of adjusting STM and CEM to specifically adapt to the new channel distributions, i.e., updating the parameters of STM, we develop denoising modules after the STM and CEM to eliminate the noise in $\bar{\boldsymbol{S}}$ and $\bar{\boldsymbol{H}}$. To further elaborate, we denote the trainable parameters of the two denoising modules as $\boldsymbol{\theta}_s$ and $\boldsymbol{\theta}_c$, respectively, and focus on the following problems:

$$\begin{aligned}&\text{channel estimation: } \boldsymbol{\theta}_{c|(\mathcal{H})} \Rightarrow \boldsymbol{\theta}_{c|\left(\bar{\boldsymbol{H}}^{(t)}\right)},\\&\text{semantic transmission: } \boldsymbol{\theta}_{s|(\mathcal{S})} \Rightarrow \boldsymbol{\theta}_{s|\left(\bar{\boldsymbol{S}}^{(t)}\right)},\end{aligned} \tag{4}$$

where $\bar{\boldsymbol{S}}^{(t)}$ and $\bar{\boldsymbol{H}}^{(t)}$ are the instantaneous data samples to be handled at time t. Note that the quality of $\bar{\boldsymbol{S}}^{(t)}$ will also be affected by the channel condition. The subscript notation "$|(\cdot)$" denotes the which data the parameter is train on, e.g., $\boldsymbol{\theta}_{c|\left(\bar{\boldsymbol{H}}^{(t)}\right)}$ denotes $\boldsymbol{\theta}_c$ is train on sample $\bar{\boldsymbol{H}}^{(t)}$. Accordingly, $\boldsymbol{\theta}_c$ is initially learned on the CSI dataset $\mathcal{H}$. Meanwhile, $\boldsymbol{\theta}_s$ is initially trained on image dataset $\mathcal{S}$ and CSI dataset $\mathcal{H}$. We treat the overall system as one entity, O2SC, and denote $\boldsymbol{\theta} = (\boldsymbol{\theta}_c, \boldsymbol{\theta}_s)$, as illustrated in Fig. 1. Thus, (4) can be simplified as

$$\text{O2SC adaptation: } \boldsymbol{\theta}_{(\mathcal{S},\mathcal{H})} \Rightarrow \boldsymbol{\theta}_{\left(\bar{\boldsymbol{S}}^{(t)},\bar{\boldsymbol{H}}^{(t)}\right)}, \tag{5}$$

where the baseline model parameters $\boldsymbol{\theta}$ are pretrained in the source domain $\mathcal{S}$ and the channel domain $\mathcal{H}$. Therefore, the main target is to adapt $\boldsymbol{\theta}_{(\mathcal{S},\mathcal{H})}$ to $\left(\bar{\boldsymbol{S}}^{t}, \bar{\boldsymbol{H}}^{(t)}\right)$. In this work, we consider an online-learning approach to make $\boldsymbol{\theta}$ adapt. Given the first image sample $\bar{\boldsymbol{S}}^{(1)}$ and channel sample $\bar{\boldsymbol{H}}^{(1)}$, the proposed framework only takes $\bar{\boldsymbol{S}}^{(1)}$ and $\bar{\boldsymbol{H}}^{(1)}$ as inputs and online update $\boldsymbol{\theta}$ to adapt to $\bar{\boldsymbol{S}}^{(1)}$ and $\bar{\boldsymbol{H}}^{(1)}$. We denote the loss function of the denoising module as $\mathcal{L}_{\text{O-L}}(\cdot)$, and the online-learning objective at $t=1$ is given by

$$\boldsymbol{\theta}^{(1)} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}_{\text{O-L}}\big(\boldsymbol{\theta}, \bar{\boldsymbol{S}}^{(1)}, \bar{\boldsymbol{H}}^{(1)}\big), \tag{6}$$

where we initialize $\boldsymbol{\theta}$ to $\boldsymbol{\theta}_{(\mathcal{S},\mathcal{H})}$. In addition, Note that only $\big(\bar{\boldsymbol{S}}^{(1)}, \bar{\boldsymbol{H}}^{(1)}\big)$ are utilized to solve (6), effectively achieving the goal of "one-shot" learning. Then, for the t-th subsequent pair of samples, $\big(\bar{\boldsymbol{S}}^{(t)}, \bar{\boldsymbol{H}}^{(t)}\big)$, our objective is to efficiently address the following optimization problem, initializing from $\boldsymbol{\theta} = \boldsymbol{\theta}^{(t-1)}$:

$$\boldsymbol{\theta}^{(t)} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}_{\text{O-L}}\big(\boldsymbol{\theta}, \bar{\boldsymbol{S}}^{(t)}, \bar{\boldsymbol{H}}^{(t)}\big). \tag{7}$$

Gradient descent is used to periodically fit the parameters of O2SC to a single pair of samples as opposed to conventional supervised learning-based methods, where a large dataset of input and labels are used. Next, we introduce how to implement O2SC with DNN models.

III. Self-Supervised Learning-Based O2SC

In this section, we begin by introducing the self-supervised learning method used to implement the previously mentioned one-shot online-learning framework. Then, we present the architecture of the proposed denoising module.

A. Self-Supervised Denoising Module

The main goal of this work is to update the system parameters online to adapt to each current data sample. In this manner, the only data used for training O2SC is the current data sample. This represents a much more challenging scenario compared to existing solutions. The key strength of this approach is that it does not rely on a large amount of training data and a long training process. In addition, it also provides better generalizability since it does not rely on training on very large data sets drawn from a specific distribution.

To realize the one-shot online-learning framework, we turn to self-supervised learning [27] and propose a plug-in denoising module. As shown in Fig. 2, we consider two denoising modules for O2SC, which are inserted after the CEM estimator and after the STM decoder. The first denoising module aims to eliminate the estimation error on $\bar{\boldsymbol{H}}$, while the latter is used to perform denoising on the initially-recovered image $\bar{\boldsymbol{S}}$. Since it is hard to conduct time-consuming end-to-end training online, we freeze the STM parameters, i.e., $(\boldsymbol{\phi}_e, \boldsymbol{\phi}_d)$, and only update the parameters of the two denoising modules, i.e., $(\boldsymbol{\theta}_c, \boldsymbol{\theta}_s)$. In general, for channel estimation, the only training data sample is the noisy initially-estimated CSI matrix $\bar{\boldsymbol{H}}$ obtained by a conventional estimator, such as the least square (LS) estimator, and there is no corresponding clean CSI matrix $\boldsymbol{H}$. For STM, the only training data sample is the noisy initially-recovered image $\bar{\boldsymbol{S}}$. The goal of the denoising module is to eliminate the noise on $\bar{\boldsymbol{H}}$ and $\bar{\boldsymbol{S}}$, and obtain a more accurate estimated CSI matrix $\hat{\boldsymbol{H}}$ and a recovered image $\hat{S}$, based on only $\bar{\boldsymbol{H}}$ and $\bar{\boldsymbol{S}}$. Since the implementations of the two denoising modules are similar, we use the notation $\boldsymbol{X}$ in the following sections to introduce the detailed designs. Thus, $\boldsymbol{X}$ should be interpreted as image $\boldsymbol{S}$ or CSI matrix $\boldsymbol{H}$. Related notation such as $\bar{\boldsymbol{X}}$ denotes the initially-estimated CSI matrix or image, depending on the context.

B. Blind Spot and Dropout

1) Two Implementation Challenges: Considering the t-th data sample $\bar{\boldsymbol{X}}^{(t)}$, the denoising module takes $\bar{\boldsymbol{X}}^{(t)}$ as input and outputs a more accurate result $\hat{\boldsymbol{X}}^{(t)}$. The training target is minimizing the mean-square error (MSE) between the estimated $\bar{\boldsymbol{X}}^{(t)}$ and the real $\boldsymbol{X}^{(t)}$, and one challenge arises. In particular, when using only the training sample $\bar{\boldsymbol{X}}^{(t)}$ ((and not the actual $\boldsymbol{X}^{(t)}$), the network will converge to an identity mapping, where the output simply mirrors the input $\bar{\boldsymbol{X}}^{(t)}$. This means the network copies the input as the output. This phenomenon occurs due to the risk of overfitting when training with a single data sample. Moreover, the denoising module can be considered as a Bayes estimator and the MSE can be decomposed as the sum of squares of estimation bias and estimation variance, where

$$\text{MSE} = \text{bias}^2 + \text{var}. \tag{8}$$

When only one noisy sample $\bar{\boldsymbol{X}}^{(t)}$ is available for training, the estimation variance will increase significantly, which also

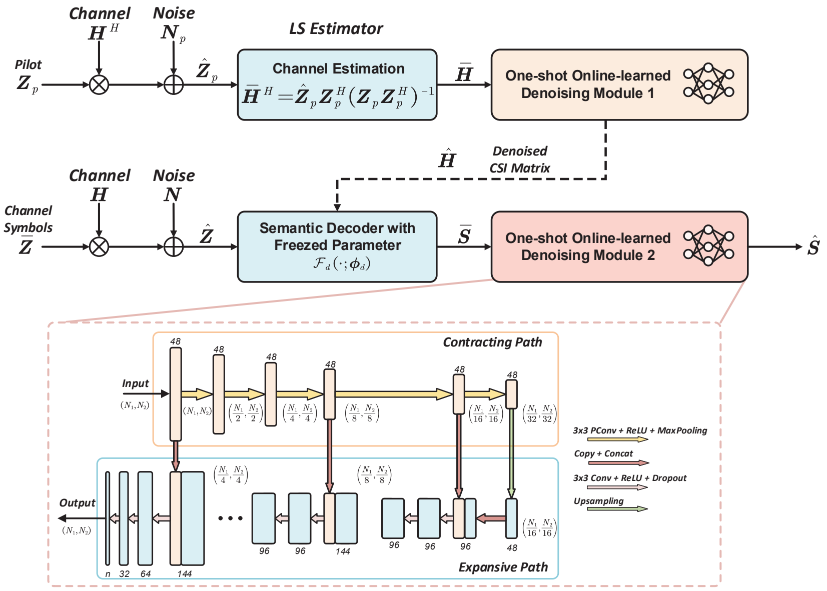

Channel
$\boldsymbol{H}^H$
Noise
$\boldsymbol{N}_p$
LS Estimator
Pilot
$\boldsymbol{Z}_p$
$\hat{\boldsymbol{Z}}_p$
Channel Estimation
$\bar{\boldsymbol{H}}^H = \hat{\boldsymbol{Z}}_p \boldsymbol{Z}_p^H (\boldsymbol{Z}_p \boldsymbol{Z}_p^H)^{-1}$
$\bar{\boldsymbol{H}}$
One-shot Online-learned Denoising Module 1
$\hat{\boldsymbol{H}}$
Denoised CSI Matrix
Channel
$\boldsymbol{H}$
Noise
$\boldsymbol{N}$
Channel Symbols
$\bar{\boldsymbol{Z}}$
$\hat{\boldsymbol{Z}}$
Semantic Decoder with Freezed Parameter
$\mathcal{F}_d(\cdot;\boldsymbol{\phi}_d)$
$\bar{\boldsymbol{S}}$
One-shot Online-learned Denoising Module 2
$\hat{\boldsymbol{S}}$
Contracting Path
Input
(N_1,N_2)
48
48
48
48
48
48
(N_1,N_2)
$\left(\frac{N_1}{2},\frac{N_2}{2}\right)$
$\left(\frac{N_1}{4},\frac{N_2}{4}\right)$
$\left(\frac{N_1}{8},\frac{N_2}{8}\right)$
$\left(\frac{N_1}{16},\frac{N_2}{16}\right)$
$\left(\frac{N_1}{32},\frac{N_2}{32}\right)$
3x3 PConv + ReLU + MaxPooling
Copy + Concat
3x3 Conv + ReLU + Dropout
Upsampling
Output
(N_1,N_2)
$\left(\frac{N_1}{4},\frac{N_2}{4}\right)$
$\left(\frac{N_1}{8},\frac{N_2}{8}\right)$
$\left(\frac{N_1}{16},\frac{N_2}{16}\right)$
n
32
64
144
96
96
144
96
96
96
48
Expansive Path

Fig. 2. The DNN structure of the denoising module. We omit superscript "(t)" in the figure for simplicity.

results in the denoised result $\hat{\boldsymbol{X}}^{(t)}$ converging to the noisy sample $\bar{\boldsymbol{X}}^{(t)}$. Thus, the key point is to avoid the identity mapping and reduce the estimation variance. To this end, we adopt the blind spot strategy [27], and apply the dropout approach in certain layers of the denoising module, which is a common regularization technique in deep learning. In the following subsections, we further discuss how to implement these approaches in detail.

2) Bernoulli Sampling-Based Blind-Spot: The blind spot theory suggests that when the network produces a specific element of a matrix, the input only provides information about the surrounding elements rather than the element itself [27]. Additionally, both the channel and image exhibit strong spatial correlations, meaning that each element in the CSI or image matrix typically has strong relationships with its surrounding elements. In the presence of spatially independent noise (or weakly-related), the denoising module using the blind spot strategy cannot isolate the noise through the surrounding elements. Instead, it can only anticipate information related to the surrounding elements. Using this observation, noise reduction can be achieved. Specifically, we perform Bernoulli sampling on the noisy sample $\bar{\boldsymbol{X}}^{(t)}$ using the blind spot strategy. A Bernoulli sampled $\check{X}_{i,j}^{(t)}$ of the i-th row j-th column in the noisy matrix $\bar{\boldsymbol{X}}^{(t)}$ with probability p is defined as

$$\check{X}_{i,j}^{(t)} = \begin{cases} \bar{X}_{i,j}^{(t)}, & \text{with probability } \ p; \\ 0, & \text{with probability } \ 1-p. \end{cases} \tag{9}$$

In each training epoch, we generate training samples by Bernoulli sampling as

$$\begin{aligned} \check{\boldsymbol{X}}_m^{(t)} &= \boldsymbol{B}_m \odot \bar{\boldsymbol{X}}^{(t)}, \\ \tilde{\boldsymbol{X}}_m^{(t)} &= (\boldsymbol{1} - \boldsymbol{B}_m) \odot \bar{\boldsymbol{X}}^{(t)}, \end{aligned} \tag{10}$$

where $m = 1, 2, \ldots, M$, M is the total number of training steps, $\boldsymbol{B}_m$ denotes the m-th binary Bernoulli matrix with element drawn from 0 or 1, and $\odot$ represents element-by-element multiplication. We denote $\tilde{\boldsymbol{X}}_m^{(t)}$ as the blind sample part that is masked by $\boldsymbol{B}_m$, and denote $\check{\boldsymbol{X}}_m^{(t)}$ as the unmasked sample part. The input to the denoising module is $\check{\boldsymbol{X}}_m^{(t)}$ and the training loss function is defined as

$$\begin{aligned} &\mathcal{L}_{\text{O-L}}\big(\boldsymbol{\theta}_x, \check{\boldsymbol{X}}^{(t)}, \tilde{\boldsymbol{X}}^{(t)}\big) = \\ &\qquad \frac{1}{M} \sum_{m=1}^{M} \|\mathcal{F}_m\big(\check{\boldsymbol{X}}_m^{(t)}; \boldsymbol{\pi}_m\big) - \tilde{\boldsymbol{X}}_m^{(t)}\|_{\boldsymbol{B}_m}^2, \end{aligned} \tag{11}$$

where $\mathcal{F}_m(\cdot; \boldsymbol{\pi}_m)$ is the model of the denoising module in the m-th training step[1], $\boldsymbol{\pi}_m$ is a vector of trainable parameters (a subset of $\boldsymbol{\theta}_x$), and $\|\boldsymbol{A}\|_{\boldsymbol{B}_m} = \|(\boldsymbol{1} - \boldsymbol{B}_m) \odot \boldsymbol{A}\|_2^2$. The loss function is measured only on the blind sample part $\tilde{\boldsymbol{X}}_m$ instead of the unmasked input $\check{\boldsymbol{X}}_m$, which is consistent with the blind spot strategy. The parameters are updated using the stochastic gradient descent (SGD).

When the noise in $\bar{\boldsymbol{X}}^{(t)}$ and $\boldsymbol{X}^{(t)}$ is independent and zero mean, the expectation of the loss function (11) is the same as that of

$$\frac{1}{M} \sum_{m=1}^{M} \|\mathcal{F}_m\big(\check{\boldsymbol{X}}_m^{(t)}; \boldsymbol{\pi}_m\big) - \boldsymbol{X}^{(t)}\|_{\boldsymbol{B}_m}^2 + \frac{1}{M} \sum_{m=1}^{M} \|\boldsymbol{\sigma}\|_{\boldsymbol{B}_m}^2, \tag{12}$$

where $\boldsymbol{\sigma}$ denotes the standard deviation of the difference between $\bar{\boldsymbol{X}}^{(t)}$ and $\boldsymbol{X}^{(t)}$. The proof is provided in [27]. Moreover, this method remains effective even when the noise is not strictly independent, since the contextual information between image pixels or between CSI element is much easier to capture than the noise itself. From (12), we see that the loss function of training with the paired samples $(\check{\boldsymbol{X}}^{(t)}, \tilde{\boldsymbol{X}}^{(t)})$ is equivalent to that of training with the clean label $\boldsymbol{X}^{(t)}$, promoting the rationality of the method.

3) Dropout for Reducing Prediction Variance: The dropout strategy aims to randomly discard some of the neural connections in a DNN model while training, where the model

[1] The model of each training step is different due to the dropout.

Algorithm 1: One-shot online-learning self-supervised training.

```
Input: The $t$-th noisy data sample (image or CSI) $\bar{\boldsymbol{X}}^{(t)}$, the
  number of training steps $M$, the number of inference
  models $T$ induced by dropout.
Output: The denoised image or channel $\hat{\boldsymbol{X}}^{(t)}$.
Training stage:
for $m = 1, 2, \ldots, M$ do
    Generate training data sample pair $\big(\check{\boldsymbol{X}}_m^{(t)}, \tilde{\boldsymbol{X}}_m^{(t)}\big)$ by
      Bernoulli sampling with equation (10).
    Perform dropout on the training model.
    Update the trainable parameters with SGD optimizer on
      loss function (12).
end
Obtain the trained model and freeze the parameters.

Inference stage:
Load the parameters trained in the first phase.
for $i = 1, 2, \ldots, D$ do
    Perform Bernoulli sampling to obtain the input $\check{\boldsymbol{X}}_i^{(t)}$.
    Perform dropout on the test model and obtain the output
      of the model $\mathcal{F}_i\big(\check{\boldsymbol{X}}_i^{(t)}; \boldsymbol{\pi}_i\big)$.
end
Average the output to obtain the denoised image or channel
  $\hat{\boldsymbol{X}}^{(t)}$.
```

structure in each training step is different. In this way, the outputs of these different induced models have a certain degree of statistical independence, which helps to reduce the variance of the predicted result. We first use the dropout strategy to generate D different induced models $\mathcal{F}_1(\cdot), \mathcal{F}_2(\cdot), \ldots, \mathcal{F}_D(\cdot)$. Then, we randomly generate D Bernoulli samples $\check{\boldsymbol{X}}_1^{(t)}, \check{\boldsymbol{X}}_2^{(t)} \ldots, \check{\boldsymbol{X}}_D^{(t)}$, which are fed into the D different models, respectively. Finally, we average the results to acquire the t-th recovered image or CSI matrix:

$$\hat{\boldsymbol{X}}^{(t)} = \frac{1}{T} \sum_{i=1}^{D} \mathcal{F}_i\big(\check{\boldsymbol{X}}_i^{(t)}; \boldsymbol{\pi}_i\big). \tag{13}$$

The one-shot self-supervised learning procedure is summarized in Algorithm 1.

C. Architecture of Proposed Denoising Module

We employ the U-Net model [35] to implement the denoising module, as depicted in Fig. 2. The denoising module consists of a contracting path and an expansive path, which are designed for extracting channel features and restoring the original resolution, respectively. The dimensions of the input are denoted as (N_1, N_2, N_3). For the CSI matrix, we split the complex-valued matrix into real and imaginary parts, and the size of the input is $(N_1, N_2, 2)$, where $N_1 = N_r$, $N_2 = N_t$, and $N_3 = 2$. For the image data sample, $N_1 = H$, $N_2 = W$, and $N_3 = 3$. Since the input of the denoising module after Bernoulli sampling can be considered as a degraded data sample, the contracting path is equipped with partial convolutional (PConv) layers. PConv layers are more effective in handling degraded signals compared to traditional convolutional layers [36]. Each PConv layer is followed by a max pooling operation with a stride of 2 for downsampling. The number of feature channels expands to 48 from N_3 after the first convolutional layer and then remains the same. The output of the contracting path is of size $(N_1/32, N_2/32, 48)$.

The expansive path is constructed with multiple blocks to reconstruct the input sample. Each block consists of two 3×3 convolutions and a concatenation operation, and each block is connected with an up-sampling layer with a scaling factor of 2. The concatenation operation combines the features of the up-sampling layer with the corresponding features from the contracting path to fuse information for improved reconstruction. The number of feature channels in each output layer of the block is 96, except for the last block. The size of output feature is restored to (N_1, N_2, N_3). Dropout is applied in each convolutional layer of the expansive path, and rectified linear units (ReLU) serve as the activation function.

D. Comparison with Supervised Learning

Here we elaborate on the differences between the proposed one-shot online-learning method and supervised methods [37]–[39] [2]. For channel denoising, supervised methods require to collect a large number of noisy data samples and corresponding clean labels as training pairs to optimize the trainable parameters of the DNN model offline. The optimization target of supervised learning-based methods can be written as

$$\boldsymbol{\theta}_x^* = \arg\min_{\boldsymbol{\theta}_x} \mathbb{E}_{(\boldsymbol{X}, \bar{\boldsymbol{X}}) \in \mathcal{X}} \left[\mathcal{L}_{\text{S-L}}\big(\boldsymbol{\theta}_x, \boldsymbol{X}, \bar{\boldsymbol{X}}\big)\right], \tag{14}$$

where $\mathcal{L}_{\text{S-L}}(\cdot)$ represents the supervised loss function, $\bar{\boldsymbol{X}}$ and $\boldsymbol{X}$ denote the noisy data sample and clean data label, respectively. In addition, $\boldsymbol{X}$ and $\bar{\boldsymbol{X}}$ are sampled from dataset $\mathcal{X}$. After offline training, the DNN model is in the actual scenarios. The supervised learning-based model performs best when the training and test environments are consistent. If the channel statistics in the test environment change, the performance will degrade due to this mismatch.

For the proposed one-shot online-learning method, we only have the noisy sample $\bar{\boldsymbol{X}}$ for training. We need to make full use of $\bar{\boldsymbol{X}}$ and generate training pairs according to $\bar{\boldsymbol{X}}$. The optimization target of our proposed self-supervised one-shot online-learning method is

$$\boldsymbol{\theta}_x^* = \arg\min_{\boldsymbol{\theta}_x} \mathbb{E}_{(\tilde{\boldsymbol{X}}_m, \check{\boldsymbol{X}}_m) \in \mathcal{B}(\bar{\boldsymbol{X}})} \left[\mathcal{L}_{\text{O-L}}\big(\boldsymbol{\theta}_x, \tilde{\boldsymbol{X}}_m, \check{\boldsymbol{X}}_m\big)\right], \tag{15}$$

where $\mathcal{B}(\cdot)$ denotes that we generate samples from the current noisy sample $\bar{\boldsymbol{X}}$, e.g., using Bernoulli sampling. Then, the trainable parameters are optimized online with the generated training pairs. In practical applications, the one-shot online-learning framework can be directly deployed online to process the real-time noisy sample.

IV. Reptile-Based Meta-Learning Algorithm

To improve the efficiency and the generalizability of the proposed framework, we incorporate a Reptile-based meta-learning approach. Specifically, we aim to find a good model

[2] Existing semantic communication methods typically rely on use of an unsupervised autoencoder since there are not explicit labels in the training phase. However, we classify general semantic communication systems as supervised learning, since the purpose of them is to construct the mapping from a noisy signal to recover the transmitted source.

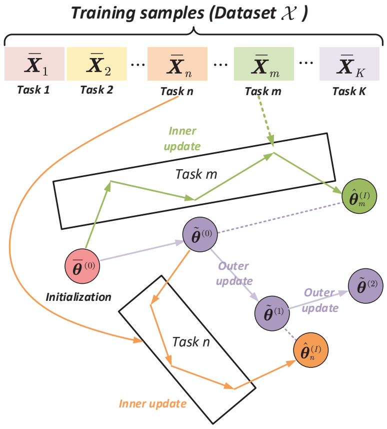

Training samples (Dataset $\mathcal{X}$)
$\bar{\boldsymbol{X}}_1$
Task 1
$\bar{\boldsymbol{X}}_2$
Task 2
$\bar{\boldsymbol{X}}_n$
Task n
$\bar{\boldsymbol{X}}_m$
Task m
$\bar{\boldsymbol{X}}_K$
Task K
Inner update
Task m
$\hat{\boldsymbol{\theta}}_m^{(I)}$
$\tilde{\boldsymbol{\theta}}^{(0)}$
$\bar{\boldsymbol{\theta}}^{(0)}$
Initialization
Outer update
$\tilde{\boldsymbol{\theta}}^{(1)}$
Outer update
$\tilde{\boldsymbol{\theta}}^{(2)}$
Task n
$\hat{\boldsymbol{\theta}}_n^{(I)}$
Inner update

Fig. 3. Illustration of the Reptile-based meta-learning scheme.

initialization for the denoising module, so that it can quickly adapt to the current environment.

A. Key Motivations for Meta-Learning

To obtain real-time results with little latency, it is essential to expedite the online training process for the denoising module. To achieve this goal, we consider employing Reptile-based meta-learning [34]. Meta-learning techniques are designed to enhance the learning capacity of neural networks by utilizing data from various distributions [40]. In the proposed framework, the parameters of the denoising module are updated based on the instantaneous data sample $\bar{\boldsymbol{X}}^{(t)}$, and we assume that the parameters converge to $\boldsymbol{\theta}_x^{(t)}$ when the training process terminates. We notice however that the convergence speed for each $\bar{\boldsymbol{X}}^{(t)}$ is highly dependent on the initialization, and a good initialization will lead to much higher training efficiency. An intuitive option for the next data sample $\bar{\boldsymbol{X}}^{(t+1)}$ is initializing $\boldsymbol{\theta}_x$ with the resulting parameters of the last sample, i.e., $\boldsymbol{\theta}_x^{(t)}$. Despite the simplicity, this approach has a severe limitation. That is, the resulting parameters $\boldsymbol{\theta}_x^{(t)}$ of the last data sample are instinctively overfitted for $\bar{\boldsymbol{X}}^{(t)}$, and may be significantly different from the expected parameters $\boldsymbol{\theta}_x^{(t+1)}$ for $\bar{\boldsymbol{X}}^{(t+1)}$. As a result, it may be hard to update $\boldsymbol{\theta}_x$ from $\boldsymbol{\theta}_x^{(t)}$ to the expected $\boldsymbol{\theta}_x^{(t+1)}$, and thus $\boldsymbol{\theta}_x^{(t)}$ may not be a suitable initialization for $\bar{\boldsymbol{X}}^{(t+1)}$. Therefore, in this work, we follow the idea of meta-learning, and aim at determining a better initialization for each instantaneous data sample, thereby enabling rapid adaptation.

B. Initialization Before Deployment

We start by finding a good initialization using some previously collected data. As shown in Fig. 3, the initial training dataset is defined as $\mathcal{X}$, which consists of K tasks, with each task serving as the basic unit of training. Within each task, the training sample is a noisy sample that could have been randomly collected in the past. For example, the training samples for CEM can be obtained at any previous time through LS channel estimation, and do not require clean channel labels. As a result, there is not much additional overhead in acquiring training samples for meta-learning.

The training process of the proposed method can be divided into two iterative phases, the inner update and the outer update. In the inner update phase, we optimize the parameters of the denoising module with one task. Our target is to find a good parameter initialization $\tilde{\boldsymbol{\theta}}_x$, such that for a randomly sampled task, the denoising module can quickly learn to deal with it. In the outer update phase, we use the obtained parameters in the inner update phase to optimize $\tilde{\boldsymbol{\theta}}_x$. Here, one task refers to the denoising operation on one data sample. We define the mapping of the denoising module as $\mathcal{G}_x(\cdot)$, with $\hat{\boldsymbol{\theta}}_x$ denoting the trainable parameters. For simplicity, we omit the subscript "x" in the following. In particular, for the denoising task k, the objective function is represented as

$$\arg\min_{\hat{\boldsymbol{\theta}}_k} \mathcal{L}_{\text{O-L}}(\mathcal{G}_x(\bar{\boldsymbol{X}}_k; \hat{\boldsymbol{\theta}}_k)),\ k = 1, \ldots, K, \tag{16}$$

where $\hat{\boldsymbol{\theta}}_k$ denotes the parameters trained with the denoising task k, and $\mathcal{L}_{\text{O-L}}(\cdot)$ is the same as the loss function (11).

The parameter update procedure is illustrated in Fig. 3, and can be summarized as follows. First, we randomly initialize $\hat{\boldsymbol{\theta}}$ and $\tilde{\boldsymbol{\theta}}$ with the same parameters $\bar{\boldsymbol{\theta}}^{(0)}$. Then, we randomly select a task m from the dataset $\mathcal{X}$ to train the denoising module. The i-th updating procedure with task m can be expressed as

$$\hat{\boldsymbol{\theta}}_m^{(i)} = \hat{\boldsymbol{\theta}}_m^{(i-1)} - \alpha \nabla_{\hat{\boldsymbol{\theta}}_m^{(i-1)}} \mathcal{L}_{\text{O-L}}(\mathcal{G}_x(\bar{\boldsymbol{X}}_m; \hat{\boldsymbol{\theta}}_m^{(i-1)})), \tag{17}$$

where α is the learning rate, $\nabla_{\hat{\boldsymbol{\theta}}_m^{(i-1)}}$ denotes the gradient with respect to $\hat{\boldsymbol{\theta}}_m^{(i-1)}$, $i = 1, \ldots, I$, and I is the total number of iterations for the update on task m. After I iterations, we finish the inner update phase on task m and finally obtain the parameters $\hat{\boldsymbol{\theta}}_m^{(I)}$.

Then, in the outer update, we compute the difference between $\hat{\boldsymbol{\theta}}_m^{(I)}$ and the expected parameter initialization $\tilde{\boldsymbol{\theta}}^{(0)}$, and use this as the direction for updating $\tilde{\boldsymbol{\theta}}^{(0)}$. Thus, the updated parameter initialization at the first step can be denoted as

$$\tilde{\boldsymbol{\theta}}^{(1)} = \tilde{\boldsymbol{\theta}}^{(0)} - \beta(\hat{\boldsymbol{\theta}}_m^{(I)} - \tilde{\boldsymbol{\theta}}^{(0)}), \tag{18}$$

where β denotes the learning rate of the outer update. The updated parameters $\tilde{\boldsymbol{\theta}}^{(1)}$ are used for the initialization of the denoising module in the next inner update phase. Then, we select another task to perform the next round of the inner and cross update. Finally, we complete the training of all the K tasks and obtain the expected initialization $\tilde{\boldsymbol{\theta}}^{(K)}$, which are used for the initialization of the denoising module for practical use. The procedure of the proposed meta-learning based initialization is summarized in Algorithm 2.

C. Online Meta-Adaptation

Although the above parameter initialization is effective when the incoming data samples are similar to those in $\mathcal{X}$, it is hard for the resulting initialization to support varying and complex channel conditions. Thus, we propose an online meta-adaptation method to simultaneously perform denoising and find a good initialization for the next data sample. In particular, we introduce a parameter initialization $\boldsymbol{\psi}_x^{(t)}$ and continuously update it with the incoming data sample $\bar{\boldsymbol{X}}^{(t)}$, so that O2SC can continuously find a good initialization for future data samples.

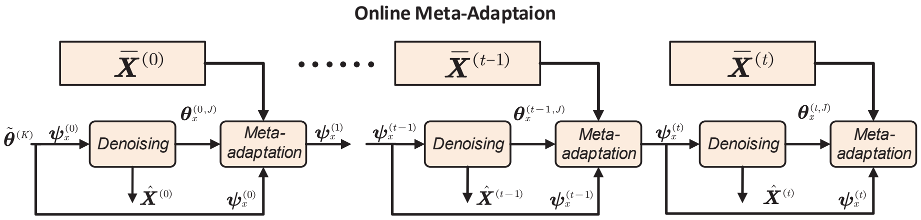

Online Meta-Adaptaion
$\bar{\boldsymbol{X}}^{(0)}$
$\bar{\boldsymbol{X}}^{(t-1)}$
$\bar{\boldsymbol{X}}^{(t)}$
$\tilde{\boldsymbol{\theta}}^{(K)}$
$\boldsymbol{\psi}_x^{(0)}$
Denoising
$\boldsymbol{\theta}_x^{(0,J)}$
Meta-adaptation
$\boldsymbol{\psi}_x^{(1)}$
$\hat{\boldsymbol{X}}^{(0)}$
$\boldsymbol{\psi}_x^{(0)}$
$\boldsymbol{\psi}_x^{(t-1)}$
Denoising
$\boldsymbol{\theta}_x^{(t-1,J)}$
Meta-adaptation
$\hat{\boldsymbol{X}}^{(t-1)}$
$\boldsymbol{\psi}_x^{(t-1)}$
$\boldsymbol{\psi}_x^{(t)}$
Denoising
$\boldsymbol{\theta}_x^{(t,J)}$
Meta-adaptation
$\hat{\boldsymbol{X}}^{(t)}$
$\boldsymbol{\psi}_x^{(t)}$

Fig. 4. Online meta adaptation and denoising procedure.

Algorithm 2: Reptile-based meta training.

```
Input: The dataset $\mathcal{X}$ for meta training, the number of tasks
  $K$, the number of iterations for inner update $I$, the learning
  rate $\alpha$, and the learning rate $\beta$ in the outer update phase.
Output: The found parameter initialization $\tilde{\boldsymbol{\theta}}^{(K)}$.
Randomly initialize the denoising module and the parameter
  initialization with parameters $\tilde{\boldsymbol{\theta}}^{(0)}$.
for $k = 1, 2, \ldots, K$ do
    Randomly select a task $k$ from the training dataset $\mathcal{X}$.
    for $i = 1, 2, \ldots, I$ do
        Compute the gradient of the loss function of task $k$.
        Update the parameters of the denoising module
          based on (17).
    end
    Update the parameter initialization $\tilde{\boldsymbol{\theta}}$ based on (18).
    Initialize the parameters of the $(k+1)$-th inner update
      with the current parameter initialization.
end
```

Algorithm 3: Online meta-adaptation.

```
Input: The current received noisy data samples
  $\bar{\boldsymbol{X}}^{(t)} (t = 1, 2, \ldots)$ for adaptation, the number of iterations
  for meta-adaptation $J$, the learning rate for meta-adaptation
  $\gamma$.
Output: The denoised result $\hat{\boldsymbol{X}}^{(t)} (t = 1, 2, \ldots)$.
Initialize $\boldsymbol{\psi}_x$ and $\boldsymbol{\theta}_x$ with the parameter initialization $\tilde{\boldsymbol{\theta}}^{(K)}$.
for $t = 1, 2, \ldots$ do
    Train the denoising module until convergence to obtain
      the denoised result $\hat{\boldsymbol{X}}^{(t)}$.
    Update the meta-adaptation parameters in the next time
      slot with $\boldsymbol{\psi}_x^{(t+1)} = \boldsymbol{\psi}_x^{(t)} - \beta\big(\boldsymbol{\psi}_x^{(t)} - \boldsymbol{\theta}_x^{(t,J)}\big)$.
    Update the parameters of the denoising module for the
      next sample with $\boldsymbol{\theta}_x^{(t+1)} = \boldsymbol{\psi}_x^{(t+1)}$.
end
```

As shown in Fig. 4, we first initialize $\boldsymbol{\psi}_x$ and $\boldsymbol{\theta}_x$ with the parameter initialization $\tilde{\boldsymbol{\theta}}^{(K)}$ learned from the aforementioned meta-learning-based initialization, i.e., $\boldsymbol{\psi}_x^{(0)} = \tilde{\boldsymbol{\theta}}^{(K)}$ and $\boldsymbol{\theta}_x^{(0)} = \tilde{\boldsymbol{\theta}}^{(K)}$. When the t-th sample $\bar{\boldsymbol{X}}^{(t)}$ is obtained, we input it to the denoising module. Then, we train the denosing module online with the one-shot online-learning method until convergence to obtain the denoised result $\hat{\boldsymbol{X}}^{(t)}$. Simultaneously, we perform the meta-adaptation procedure to find a good initialization for $\bar{\boldsymbol{X}}^{(t+1)}$. In particular, the parameters of the denoising module in the j-th test step are updated as

$$\boldsymbol{\theta}_x^{(t,j)} = \boldsymbol{\theta}_x^{(t,j-1)} - \gamma \nabla_{\boldsymbol{\theta}_x^{(t,j-1)}} \mathcal{L}_{\text{O-L}}\big(\mathcal{G}_x(\bar{\boldsymbol{X}}^{(t)}; \boldsymbol{\theta}_x^{(t,j-1)})\big), \quad (19)$$

where γ is the learning rate, $j = 1, \ldots, J$, and J is the number of iterations. We use parameters $\boldsymbol{\psi}_x^{(t)}$ for meta-adaptation, as given by

$$\boldsymbol{\psi}_x^{(t+1)} = \boldsymbol{\psi}_x^{(t)} - \beta\big(\boldsymbol{\psi}_x^{(t)} - \boldsymbol{\theta}_x^{(t,J)}\big). \quad (20)$$

Then, we exploit $\boldsymbol{\psi}_x^{(t+1)}$ to initialize the parameters of the denoising module for the $(t+1)$-th data sample as

$$\boldsymbol{\theta}_x^{(t+1)} = \boldsymbol{\psi}_x^{(t+1)}. \quad (21)$$

As the number of samples for adaptation increases, the training of the denoising module will converge faster when dealing with new data samples, which improves the efficiency of online training. The overall procedure is summarized in Algorithm 3.

V. Simulation Results

In this section, we first outline the experimental configuration and evaluation protocols. Next, we verify the performance of the proposed one-shot online-learning algorithms via simulations.

A. Simulation Setup

1) Datasets and Channel Models: In the simulations, we consider a transmitter equipped with different numbers of antennas. We generate the channels with the “3GPP-3D” model in the QuaDRiGa simulation platform, and we also consider the narrowband millimeter wave (mmWave) clustered channel model [41]. For the mmWave channel case, the numbers fo clusters and rays are set to 4 and 3, respectively. We implement the proposed one-shot online-learning scheme with the deep learning platform “Pytorch”. The “Adam” optimizer is employed, with a batch size of 128. We use the CIFAR10 dataset that consists of $50,000$ color images of size $32 \times 32 \times 3$ in the training dataset and $10,000$ images in the test dataset. We also carried out some experiments on higher-resolution images, using the STL-10 dataset consisting of $96 \times 96 \times 3$ images. Here we evaluated the end-to-end processing latency of these schemes. The experiments were implemented on PyTorch 1.9.1 using an Intel(R) Core(TM) i5-10400 CPU and an RTX 3060 GPU. The task number K for meta adaptation is set to 50. In addition, we take the DJSCC model [16] as the architecture for the encoder and decoder. To improve its performance, we add the precoding and combiner layer to train the model, and jointly train the DJSCC with different channel realizations. Moreover, the advanced SNR-adaptive scheme, ADJSCC [17], is also considered. We also augment it with the precoding and combiner layer.

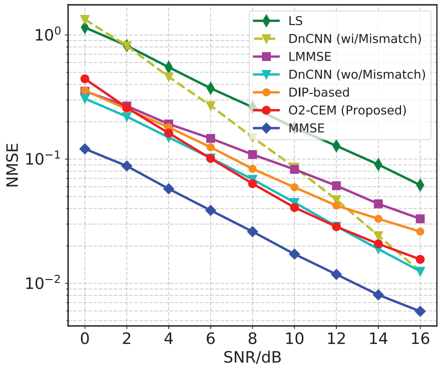


Fig. 5. NMSE performance versus SNR.

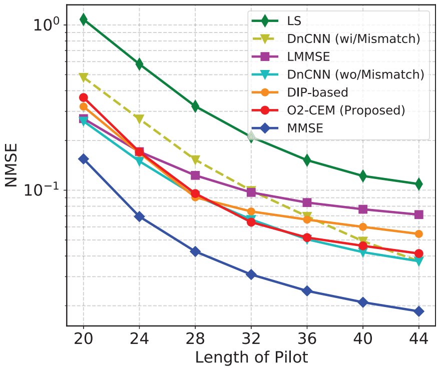


Fig. 6. NMSE performance versus the length of pilots.

2) Evaluation Metrics: To evaluate the performance of the proposed O2SC, peak signal-to-noise ratio (PSNR) is chosen for distortion metric. It measures the ratio between the maximum possible power and the noise, and can be calculated by

$$\mathrm{PSNR} = 10 \log_{10} \frac{\mathrm{MAX}^2}{\mathrm{MSE}} (\mathrm{dB}), \tag{22}$$

where MAX is the maximum possible value of the pixels, e.g., MAX equals 255 for images in RGB format. Normalized mean squared error (NMSE) is used as the metric for channel estimation, where

$$\mathrm{NMSE} = \mathbb{E}_{\hat{\boldsymbol{H}},\boldsymbol{H}} \left[\|\hat{\boldsymbol{H}} - \boldsymbol{H}\|_2^2 / \|\boldsymbol{H}\|_2^2 \right]. \tag{23}$$

3) Comparison Schemes: For comparison, we adopt BPG source coding and low-density parity-check (LDPC) channel coding together with 4-order quadrature amplitude modulation. (QAM) In addition, we adopt the channel coding scheme in the 802.11n protocol [42], [43], and test the performance of the BPG+LDPC approach with the following two code rates: $1/2$ and $2/3$. The channel bandwidth ratios are selected as $1/16$ and $1/48$, and d is set to 4.

Moreover, we also present the effectiveness of our proposed approach via separate ablation studies. Thus, the performance of the CEM is also considered. In particular, we compare the performance of the proposed one-shot online-learning method with that of the linear minimum mean square error (LMMSE) channel estimator, and the DnCNN-based supervised learning method [44]. The DnCNN-based methods are trained with $10,000$ channel samples to achieve satisfactory performance. The deep image prior (DIP)-based channel estimation method is also included as a benchmark [45]. We use the MMSE estimator as a performance bound.

B. Ablation Study

In this subsection, we show the superiority of the one-shot online-learning framework by conducting ablation studies. We test the corresponding performance gain of the denoising modules on CEM and STM, and the related results are labeled as O2-CEM and O2-STM in the following simulation results. Furthermore, we also jointly consider the two modules and present the overall performance gain at last.

1) Channel Estimation Performance Comparison: Fig. 5 illustrates the NMSE performance of the proposed one-shot online-learning framework and the benchmarks for different values of SNR. For "DnCNN (wo/Mismatch)", the training and test SNR configurations are the same; for "DnCNN (wi/Mismatch)", we train the DnCNN assuming SNR $= 18$ dB and test it for other SNRs. We observe that the performance of the proposed algorithm is much better than that of the LS estimator, indicating the effectiveness of the denoising module. In addition, in the high SNR regime, the performance of our framework is better than that of the DIP-based method and the LMMSE estimator. Furthermore, the performance of our self-supervised learning framework is very close to that of the "DnCNN (wo/Mismatch)" and is better than that of "DnCNN (wi/Mismatch)", which illustrates that our approach is more generalizable than supervised learning approaches. The performance gap increases as the SNR decreases, since the mismatch between the training and test stage increases for the supervised learning.

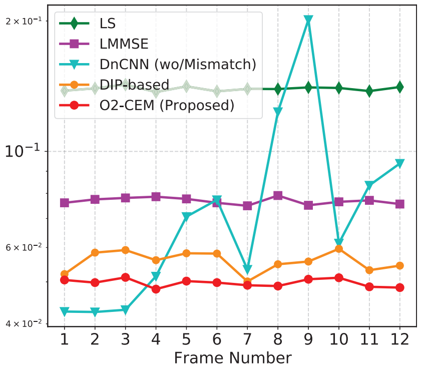
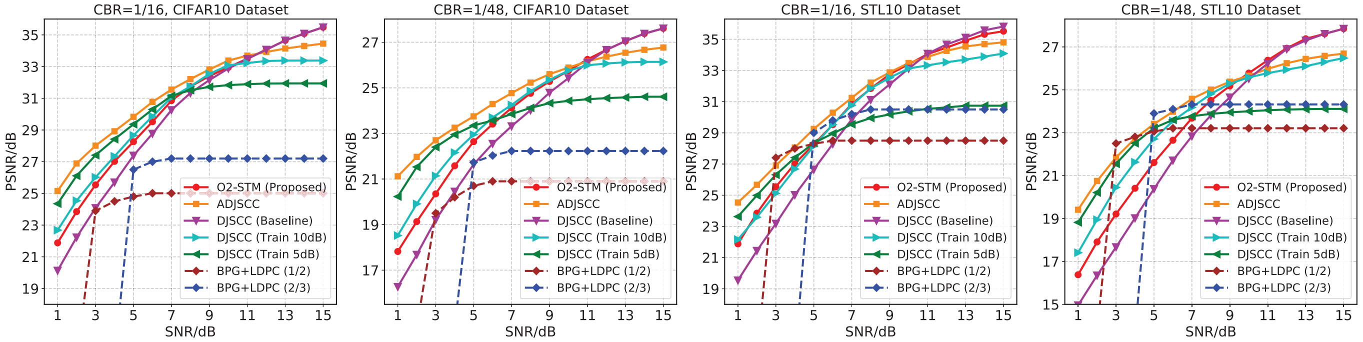


Fig. 7. NMSE performance versus the number of data frames.

Fig. 6 illustrates the NMSE performance of the proposed one-shot online-learning framework and the benchmarks for different pilot lengths. For "DnCNN (wo/Mismatch)", the pilot length L in the training and test stage are the same; while for "DnCNN (wi/Mismatch)", we train DnCNN with $L = 44$ and test it with other pilot lengths. It can be seen that the estimation performance of O2-CEM is much better than that of the LS estimator with the same length of pilots. The NMSE of O2-CEM with 24 pilots is better than that of the LS estimator with 44 pilots, which demonstrates that our proposed channel estimation method can reduce pilot overhead. In addition, our



(a)

(b)

(c)

(d)

Fig. 8. PSNR performance comparison versus average received channel SNR over the mmWave channel. The baseline model is trained with error-free transmission, and the other schemes are trained at SNR $= 5$ dB and SNR $= 10$ dB. ADJSCC is trained with SNR sampled from 0 dB to 11 dB. The plots in (a) and (b) show the results achieved in CIFAR10 dataset at CBR $= 1/16$ and $1/48$, respectively. The plots in (c) and (d) present the results achieved in STL10 dataset with higher resolution at CBR $= 1/16$ and $1/48$, respectively.

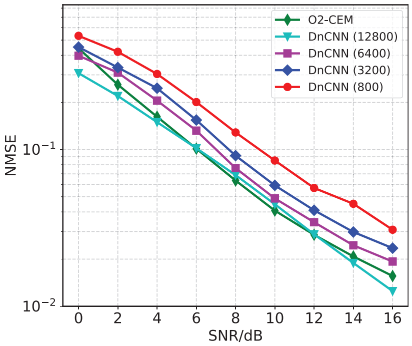


Fig. 9. NMSE performance comparisons when different number of data points are utilized by the supervised learning-based scheme.

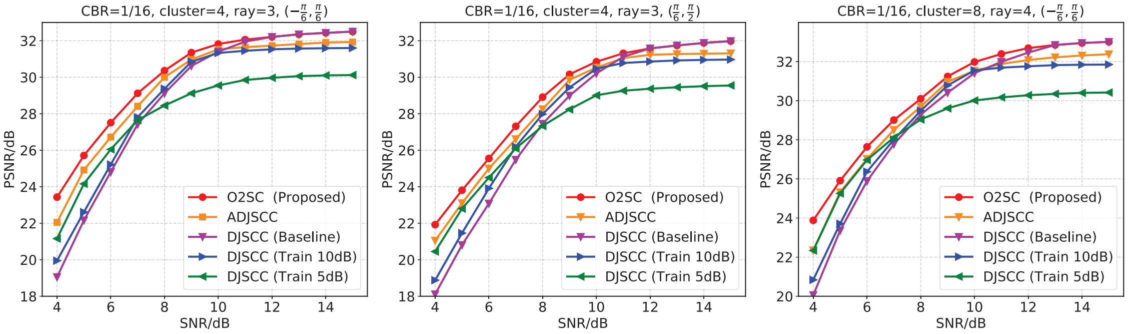
one-shot learning technique achieves comparable performance with the ‘DnCNN (wo/Mismatch)’ based supervised learning approach. When there is a mismatch in the number of pilot symbols in the training and test stages, our method is much better than supervised learning, which demonstrates that our framework can adapt to different pilot lengths.

Fig. 7 illustrates the NMSE performance of the proposed one-shot online-learning framework and the benchmarks when the channel changes with the frame. We simulate the channel variation by changing the receiver’s position. In the first three frames, we keep the receiver’s position fixed; the channel remains unchanged and it is consistent with the training channels for supervised learning. In this case, DnCNN performs well. From the 4-th frame, we fix the position of BS and move the receiver, which causes changes to angle of arrival (AoA) of the line-of-sight (LOS) path. We can see that the performance of the DnCNN based supervised learning method deteriorates due to the mismatch between the training and application environments. At the 8-th and 9-th frame, the performance of DnCNN deteriorates dramatically, which is likely because because the test channel is no significantly different from the training channels. In contrast, the performance of our proposed self-supervised learning method remains stable, which illustrates its ability to adapt to variable channels in dynamic environments.

In the next example, we compare the number of learning shots by varying the number of training CSI samples for the supervised learning-based benchmarks and compare its performance with the proposed one-shot method. The results are provided in Fig. 9. It can be observed that while training with more data points can indeed improve the estimation performance of the supervised-learning-based DnCNN, this is challenging to achieve in real communication systems. In comparison, our proposed O2-CEM exhibits comparable performance without requiring a large number of training data samples. This further demonstrates the superiority of the proposed online-learning scheme.

2) Image Transmission Performance: Fig. 8 illustrates the performance of the proposed one-shot online-learning method for STM under different average received SNR values and different channel bandwidth ratios (CBR). To isolate the contribution of O2-CEM, we initially assume perfect CSI estimation, rendering the performance gain of O2-CEM irrelevant. The legend “DJSCC (Baseline)” denotes training the DJSCC model in an error-free manner. In addition, we also train the DJSCC model at SNR $= 5$ dB and SNR $= 10$ dB, and label the results as “DJSCC (Train 5 dB)” and “DJSCC (Train 10 dB)”, respectively. For the proposed method, we implement O2-STM by inserting the one-shot online-learning denoising module after the decoder of the baseline model, and label the results as “O2-STM (Proposed)”. This also ensures that the O2-STM has no channel information before deployment, since the baseline model is trained in an error-free manner. Then, we test the above schemes on the test images for varying SNR values.

Fig. 8(a)-(b) depicts the PSNR performance on the CIRAR10 datasets for bandwidth compression ratios CBR $= 1/16$ and CBR $= 1/48$. We see that the deep learning-based methods outperform the traditional separated BPG+LDPC approach, which suffers from a cliff-effect when the channel condition is below the level anticipated by the channel code. Additionally, we observe that the model trained at relatively low SNR achieves better performance in low SNR regimes, while the model trained at higher SNR tends to achieve better performance. Specifically, when the test SNR exceeds the training SNR, the performance quickly saturates. This



(a) (b) (c)

Fig. 10. PSNR performance comparison versus average received channel SNR over different channel environments with varying numbers of paths and different angles.

indicates that there is an inherent trade-off between the amount of error protection and compression, as also illustrated in [16]. In contrast, when the SNR is relatively high, our proposed method significantly outperforms the models trained under a specific SNR, even though our proposed approach is not trained under a specific SNR condition. This is mainly because the proposed method is based on the baseline model, which mainly allocates all amount of symbols for compression. The proposed method exploits the characteristics of the noise to further improve its performance. Moreover, we find that the performance gain of the proposed method increases as the SNR decreases and nearly approaches the performance of the model trained at SNR $= 10$ dB. In addition, compared with the SNR-adaptive scheme ADJSCC, teh proposed method can approach its performance at higher SNR, even without training with a number of channel samples and SNR like ADJSCC. By comparing Fig. 8(a) and (b), we see that the denoising module is still effective in more severe scenarios with lower transmission bandwidth. Furthermore, it is worth noting that despite the slight performance loss compared with the DJSCC model trained at a specific SNR, the ability of O2-STM to dynamically adapt to varying communication conditions without requiring time-consuming offline training provides a significant advantage in practical wireless image transmission scenarios.

We also evaluate the proposed O2-STM scheme on a higher resolution image dataset, STL-10. The results are shown in Fig. 8(c)-(d).We see from the figures that the proposed method is still effective and achieves comparable performance to the model trained in low SNR regimes, and significantly outperforms other methods in high SNR regimes. In addition, since the traditional methods are specifically designed for high-resolution images, they achieve better performance than the deep learning-based methods at the endpoint of the cliff, i.e., SNR $= 3$ dB or SNR $= 5$ dB. However, this may be addressed by adopting a carefully-designed model with more parameters.

We further evaluate the overall performance of O2SC by incorporating both channel estimation and semantic transmission, and the results are shown in Fig. 10 for the case of the mmWave channel model. We observe that the proposed O2SC method significantly outperforms the ADJSCC and the DJSCC models trained at SNR $= 5$ dB, SNR $= 10$ dB, and with error-free transmission, especially in low SNR regimes. This is mainly because the denoising modules are more effective when dealing with higher levels of noise. This highlights the advantages of jointly considering channel estimation in semantic communication and suggests the potential for a joint end-to-end design for practical implementations. Here we further include the performance of semantic transmission under the mmWave channel model with different numbers of clusters and rays, and we also investigate the impact of the azimuth AoA and AoD. The performance gains observed in different channel environments further demonstrate the significant generalizability of the proposed approach, making it a potential solution for realizing channel-adaptive transmission in future semantic communication systems.

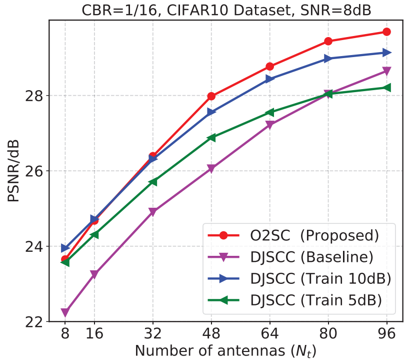


Fig. 11. Overall PSNR performance of O2SC for different numbers of transmit antennas.

Fig. 11 shows the PSNR performance for different numbers of transmit antennas, where the number of received antennas is set to $N_r = 8$ and the test SNR is set to 5 dB. We respectively train the benchmarks at specific settings, where a number of collected channel samples are required by the models trained at SNR $= 5$ dB and SNR $= 10$ dB. We find that the performance of the proposed method improves with the number of transmit antennas. This result also implicitly shows the generalizability of the proposed method under different system settings, which is an important property in practical settings. Additionally, it is evident that our proposed method outperforms the benchmarks. Furthermore, we can observe that the performance gap between the proposed O2SC and

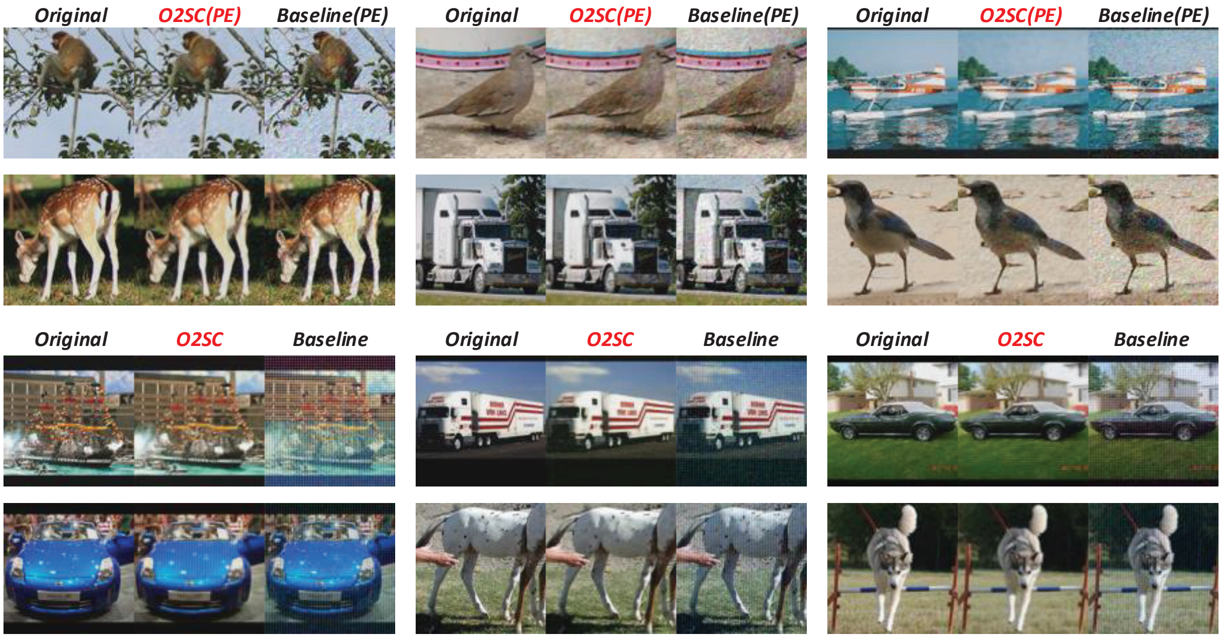


Fig. 12. Examples of output images for visual comparison. For each triplet, the left is original image, the medium is the reconstructed image using O2SC, and the right is the reconstructed image by using the baseline model. "(PE)" represents using perfect channel estimation.

specifically-trained models (i.e., the models trained at SNR $= 5$ dB and SNR $= 10$ dB) becomes larger as N_t increases, which further illustrates the performance advantage.

To further demonstrate the effectiveness of the proposed framework, we also conduct visual studies, and the results are shown in Fig. 12. In the figure, "(PE)" represents using perfect channel estimation, so that the influence of estimation error can be escaped for the ablation requirement. We observe that wireless transmission noise causes disturbances in the reconstructed image, significantly reducing the visual quality. However, with the assistance of the proposed denoising module, the visual quality can be significantly improved. This further demonstrate that, though we do not considered channel condition when encoding the image at the transmitter, it is still able to eliminate the transmission error based on the correlation between different image parts.

C. Convergence Analysis

We evaluate the online training latency of the proposed methods by plotting the convergence curve in Fig. 13(a). We calculate the loss of online training for the first sample as a function of the number of iterations. The red curve represents initializing the O2SC with meta-learning, while the purple curve represents random initialization. As shown, O2SC converges much faster using meta-learning-based approach. In addition, the loss at convergence for the meta-learning-based approach is much lower. This observation indicates that the meta-learning-based method significantly improves the convergence speed, and achieves a higher efficiency. In addition, the results also demonstrate that the meta-learning-based initialization can also yield significant performance gain compared to using random initialization.

To further explore the impact of K, we compare the convergence speed with varying numbers of channel samples. The results, shown in Fig. 13(b), indicate that the convergence speed increases with K, demonstrating the effectiveness of the meta-learning-based methods. It can also be seen that the resulting loss decreases with the number of tasks. Additionally, we observe that even a small value of K significantly speeds up convergence. This finding suggests that the proposed method does not rely on a large number of samples and can perform effectively with only a few samples.

We examined the impact of the learning rate, as shown in Fig. 13(c). It is evident that a higher learning rate accelerates convergence but also causes a slight degradation in performance. Therefore, there is a trade-off between convergence speed and estimation accuracy. In practical applications, selecting an appropriate learning rate is crucial for balancing these factors.

D. Complexity Analysis

The proposed O2SC framework consists of two key components: STM and CEM. The STM primarily incorporates a CNN-based JSCC model and a U-Net-based image denoising module, while the CEM is also predominantly built on U-Net architecture. Concretely, the computational cost (multiply-accumulate operations) of a single convolutional layer is given by $\mathcal{O}(K^2 C_i C_o H_o W_o)$, where K denote the kernel size, C_i denotes the input channels, and C_o denotes the output channels. Moreover, H_o and W_o denotes the height and width of the intermediate feature, respectively. As a result, the computational complexity of the CNN-based DJSCC model can be written as $\mathcal{O}\left(\sum_{l=1}^{L_1} C_{1,i,l} C_{1,o,l} K_{1,l}^2 H_{1,o,l}, W_{1,o,l}\right)$, where the subscript "$l$" signifies the layer number. For example, $K_{1,l}$ denotes the kernel size of l-th layer. Since $H_{o,l}$ and $W_{o.l}$ are only related to the size of the input images, (H, W), the computational complexity can be further simplified as

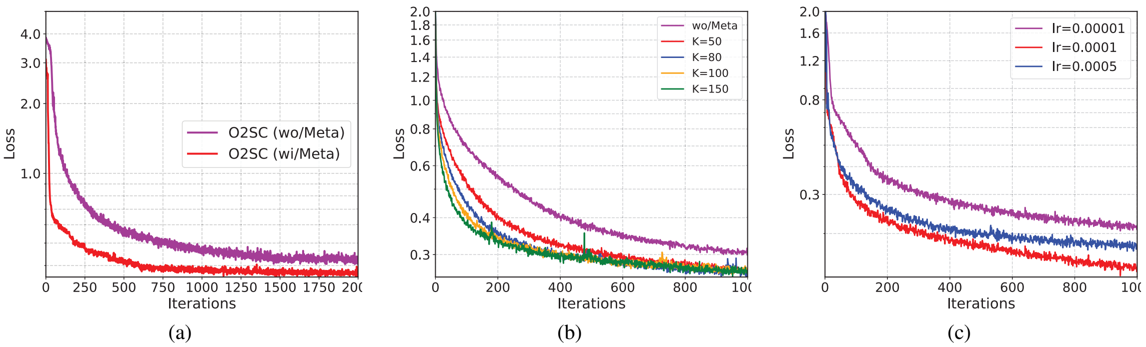


Fig. 13. Convergence analysis. (a) Convergence curve of O2SC with/without meta-learning; (b) Convergence curve of O2SC using different numbers of tasks K, where we consider $K = 0, 50, 80, 100, 150$, respectively; (c) Convergence curve of O2SC using different learning rates.

$\mathcal{O}\left(HW\sum_{l=1}^{L_1} C_{1,i,l}C_{1,o,l}K_{1,l}^2\right)$. Meanwhile, the computational complexity of the U-Net in STM can be represented as $\mathcal{O}\left(N_s HW\sum_{l=1}^{L_2} C_{2,i,l}C_{2,o,l}K_{2,l}^2\right)$, where N_s denotes the number of training iterations, and L_2 is the number of layers in STM. Similar to STM, the computational complexity of CTM can be represented as $\mathcal{O}\left(N_c N_r N_t\sum_{l=1}^{L_3} C_{3,i,l}C_{3,o,l}K_{3,l}^2\right)$, where N_c denotes the number of training iterations for CTM, and (N_r, N_t) determines the size of the channel matrix. Considering the complexity of LS estimator, the overall complexity of channel estimation can be expressed as $\mathcal{O}\left(N_r^2 N_t^2 + N_c N_r N_t\sum_{l=1}^{L_3} C_{3,i,l}C_{3,o,l}K_{3,l}^2\right)$.

E. Discussion

It has been recognized that channel fading is also a key factor affecting the performance of semantic communication. For existing supervised learning-based semantic communication, the semantic encoder and decoder are jointly learned under a number of collected channel samples in an offline manner before deployment. This helps capture the fading effect, thereby enhancing performance. However, the availability of samples is generally rare in practical communication systems, and the channels are time-varying, so it is difficult to use fixed models trained offline to support real-time applications. Regarding the above experimental results and ablation studies, we can conclude two advantages of the proposed O2SC:

- The one-shot online-learning framework is highly adaptable to different channel conditions without collecting a large number of training samples.
- The one-shot online-learning framework can be directly deployed online to process real-time received noisy samples. Thus, the proposed self-supervised learning approach can adapt to varying channels in dynamic environments.

VI. Conclusion

In this paper, we proposed a new method, referred to as O2SC, that focuses on the joint problem of channel estimation and semantic transmission to achieve channel-adaptive semantic communication systems. This approach can be directly deployed online without the need for acquiring a large number of channel samples for offline training, making it robust to varying channels in dynamic environments. The framework is implemented by devising a plug-in denoising module. The denoising module is designed based on the one-shot learning method, where Bernoulli sampling and dropout approaches are employed. To accelerate the online learning, the meta-adaptation mechanism was proposed to find a good initialization for different channel samples. Simulation results showed that our proposed one-shot online-learning scheme achieves performance comparable to supervised learning methods and has better generalizability. The proposed online-learning paradigm intrinsically provides a new solution for achieving adaptive semantic communication.

We introduced a one-shot online learning framework to enhance the performance of semantic communication systems, focusing initially on the problem formulation. This innovative approach enables rapid adaptation to new data samples in dynamic environments, resulting in significant performance gains. However, several directions show promise for further improvement. First, limited attention has been given to the network design. Developing efficient and robust network architectures is critical for optimizing the performance of the online learning framework. Secondly, investigating the impact of shifts in source and channel domains is essential. These shifts can significantly influence performance and generalizability. Addressing these areas will be crucial for advancing the capabilities and applicability of semantic communication systems in real-world scenarios. We acknowledge these limitations and leave them for future research.

References

[1] C. E. Shannon, “A mathematical theory of communicatio,” *Bell Syst. Tech. J.*, vol. 27, pp. 379–423, Jul. 1948.

[2] K. Niu, J. Dai, S. Yao, S. Wang, Z. Si, X. Qin, and P. Zhang, “A paradigm shift toward semantic communications,” *IEEE Comm. Mag.*, vol. 60, no. 11, pp. 113–119, Nov. 2022.

[3] G. Shi, Y. Xiao, Y. Li, and X. Xie, “From semantic communication to semantic-aware networking: Model, architecture, and open problems,” *IEEE Commun. Mag.*, vol. 59, no. 8, pp. 44–50, Aug. 2021.

[4] J. Huang, D. Li, C. Huang, X. Qin, and W. Zhang, “Joint task and data oriented semantic communications: A deep separate source-channel coding scheme,” *IEEE Internet Things J.*, to appear, 2023, doi: 10.1109/JIOT.2023.3293154.

[5] D. Gündüz, Z. Qin, I. E. Aguerri, H. S. Dhillon, Z. Yang, A. Yener, K. K. Wong, and C.-B. Chae, “Beyond transmitting bits: Context, semantics, and task-oriented communications,” *IEEE J. Sel. Areas Commun.*, vol. 41, no. 1, pp. 5–41, Jan. 2023.

[6] Q. Hu, G. Zhang, Z. Qin, Y. Cai, G. Yu, and G. Y. Li, “Robust semantic communications with masked VQ-VAE enabled codebook,” *IEEE Trans. Wireless Comm.*, vol. 22, no. 12, pp. 8707–8722, Dec. 2023.

[7] H. Xie, Z. Qin, G. Y. Li, and B. Juang, "Deep learning enabled semantic communication systems," *IEEE Trans. Signal Process.*, vol. 69, pp. 2663–2675, Apr. 2021.

[8] T. Wu, Z. Chen, D. He, L. Qian, Y. Xu, M. Tao, and W. Zhang, "CDDM: Channel denoising diffusion models for wireless communications," *arXiv preprint arXiv:2210.15347*, 2023.

[9] 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, Jan. 2023.

[10] K. Matsumoto, Y. Inoue, Y. Hara-Azumi, K. Maruta, Y. Nakayama, and D. Hisano, "Impact of quantization noise on CNN-based joint source-channel coding and modulation," in *IEEE Consumer Commun. Netw. Conf. (CCNC)*, Aug. 2023, pp. 465–468.

[11] S. Wang, J. Dai, Z. Liang, K. Niu, Z. Si, C. Dong, X. Qin, and P. Zhang, "Wireless deep video semantic transmission," *IEEE J. Select. Areas Commun.*, vol. 41, no. 1, pp. 214–229, Jan. 2023.

[12] H. Hu, X. Zhu, F. Zhou, W. Wu, R. Q. Hu, and H. Zhu, "One-to-many semantic communication systems: Design, implementation, performance evaluation," *IEEE Commun. Lett.*, vol. 26, no. 12, pp. 2959–2963, Dec. 2022.

[13] X. Mu and Y. Liu, "Exploiting semantic communication for non-orthogonal multiple access," *IEEE J. Select. Areas Commun.*, vol. 41, no. 8, pp. 2563–2576, Aug. 2023.

[14] H. Yoo, L. Dai, S. Kim, and C.-B. Chae, "On the role of ViT and CNN in semantic communications: Analysis and prototype validation," *IEEE Access*, vol. 11, pp. 71 528–71 541, Jul. 2023.

[15] P. Jiang, C.-K. Wen, S. Jin, and G. Y. Li, "Deep source-channel coding for sentence semantic transmission with HARQ," *IEEE Trans. Commun.*, vol. 70, no. 8, pp. 5225–5240, Aug. 2022.

[16] E. Bourtsoulatze, D. Burth Kurka, and D. Gündüz, "Deep joint source-channel coding for wireless image transmission," *IEEE Trans. Cognit. Comm. Netw.*, vol. 5, no. 3, pp. 567–579, Sep. 2019.

[17] J. Xu, B. Ai, W. Chen, A. Yang, P. Sun, and M. Rodrigues, "Wireless image transmission using deep source channel coding with attention modules," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 4, pp. 2315–2328, Apr. 2022.

[18] M. Yang and H.-S. Kim, "Deep joint source-channel coding for wireless image transmission with adaptive rate control," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, May 2022, pp. 5193–5197.

[19] W. Zhang, H. Zhang, H. Ma, H. Shao, N. Wang, and V. C. M. Leung, "Predictive and adaptive deep coding for wireless image transmission in semantic communication," *IEEE Trans. Wireless Commun.*, vol. 22, no. 8, pp. 5486–5501, Aug. 2023.

[20] P. Jiang, C.-K. Wen, S. Jin, and G. Y. Li, "Wireless semantic transmission via revising modules in conventional communications," *IEEE Wireless Commun.*, vol. 30, no. 3, pp. 28–34, Jun. 2023.

[21] M. Yang, C. Bian, and H.-S. Kim, "OFDM-guided deep joint source channel coding for wireless multipath fading channels," *IEEE Trans. Cognit. Comm. Netw.*, vol. 8, no. 2, pp. 584–599, Jun. 2022.

[22] H. Wu, Y. Shao, C. Bian, K. Mikolajczyk, and D. Gündüz, "Vision transformer for adaptive image transmission over MIMO channels," *arXiv preprint arXiv:2210.15347*, 2022.

[23] S. Yao, S. Wang, J. Dai, K. Niu, and P. Zhang, "Versatile semantic coded transmission over MIMO fading channels," *arXiv preprint arXiv:2210.16741*, 2022.

[24] G. Zhang, Q. Hu, Y. Cai, and G. Yu, "Scan: Semantic communication with adaptive channel feedback," *IEEE Trans. Cognit. Comm. Netw.*, pp. 1–1, to appear, 2024, doi: 10.1109/TCCN.2024.3394867.

[25] J. Park, Y. Oh, S. Kim, and Y.-S. Jeon, "Joint source-channel coding for channel-adaptive digital semantic communications," *IEEE Trans. Cognit. Comm. Netw.*, pp. 1–1, to appear, 2024, doi: 10.1109/TCCN.2024.3422496.

[26] N. Ginige, K. B. Shashika Manosha, N. Rajatheva, and M. Latva-aho, "Untrained DNN for channel estimation of RIS-assisted multi-user OFDM system with hardware impairments," in *Annu. Int. Symp. Pers. Indoor Mobile Radio Commun.(PIMRC)*, Oct. 2021, pp. 561–566.

[27] Y. Quan, M. Chen, T. Pang, and H. Ji, "Self2Self with dropout: Learning self-supervised denoising from single image," Jun. 2020, p. 1887–1895.

[28] S. Thrun and L. Pratt, "Learning to learn: Introduction and overview," *Boston, MA, USA: Springer*, pp. 3–17, 1998.

[29] Y. Yuan, G. Zheng, K.-K. Wong, B. Ottersten, and Z.-Q. Luo, "Transfer learning and meta learning-based fast downlink beamforming adaptation," *IEEE Trans. Wireless Commun.*, vol. 20, no. 3, pp. 1742–1755, Nov. 2021.

[30] J. Zhang, Y. Yuan, G. Zheng, I. Krikidis, and K.-K. Wong, "Embedding model-based fast meta learning for downlink beamforming adaptation," *IEEE Trans. Wireless Commun.*, vol. 21, no. 1, pp. 149–162, Jan. 2022.

[31] O. Simeone, S. Park, and J. Kang, "From learning to meta-learning: Reduced training overhead and complexity for communication systems," in *2020 2nd 6G Wireless Summit (6G SUMMIT)*, 2020.

[32] I. Nikoloska and O. Simeone, "Modular meta-learning for power control via random edge graph neural networks," *IEEE Trans. Wireless Commun.*, vol. 22, no. 1, pp. 457–470, Jan. 2023.

[33] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," *Proc. 34th Int. Conf. Mach. Learn. (ICML)*, vol. 70, pp. 1126–1135, Aug. 2017.

[34] A. Nichol, J. Achiam, and J. Schulman, "On first-order meta-learning algorithms," *arXiv preprint arXiv:1803.02999*, 2018.

[35] Z. Gu, J. Cheng, H. Fu, K. Zhou, H. Hao, Y. Zhao, T. Zhang, S. Gao, and J. Liu, "CE-Net: Context encoder network for 2D medical image segmentation," *IEEE Trans. Med. Imag.*, vol. 38, no. 10, pp. 2281–2292, Oct. 2019.

[36] G. Liu, F. A. Reda, K. J. Shih, T.-C. Wang, A. Tao, and B. Catanzaro, "Image inpainting for irregular holes using partial convolutions," *Eur. Conf. Comput. Vis. (ECCV)*, pp. 85–100, Sep. 2018.

[37] P. Dong, H. Zhang, G. Y. Li, I. S. Gaspar, and N. NaderiAlizadeh, "Deep CNN-based channel estimation for mmwave massive MIMO systems," *IEEE J. Sel. Topics Signal Process.*, vol. 13, no. 5, pp. 989–1000, Sep. 2019.

[38] M. B. Mashhadi and D. Gündüz, "Pruning the pilots: Deep learning-based pilot design and channel estimation for MIMO-OFDM systems," *IEEE Trans. Wireless Commun.*, vol. 20, no. 10, pp. 6315–6328, 2021.

[39] X. Ma and Z. Gao, "Data-driven deep learning to design pilot and channel estimator for massive mimo," *IEEE Trans. Veh. Technol.*, vol. 69, no. 5, pp. 5677–5682, May 2020.

[40] Y. Yuan, G. Zheng, K.-K. Wong, and K. B. Letaief, "Meta-reinforcement learning based resource allocation for dynamic V2X communications," *IEEE Trans. Veh. Tech.*, vol. 70, no. 9, pp. 8964–8977, Sep. 2021.

[41] S. S. Ioushua and Y. C. Eldar, "A family of hybrid analog–digital beamforming methods for massive MIMO systems," *IEEE Trans. Signal Process.*, vol. 67, no. 12, pp. 3243–3257, Jan. 2019.

[42] N. K. Ahmed Mahdi and V. Paliouras., "A multirate fully parallel LDPC encoder for the IEEE 802.11 n/ac/ax QC-LDPC codes based on reduced complexity XOR trees." *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 29, no. 1, pp. 51–64, Apr. 2020.

[43] 3GPP, "NR; physical layer procedures for data," 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 38.214, 2018, version 15.0.0.

[44] S. Liu, Z. Gao, J. Zhang, M. D. Renzo, and M.-S. Alouini, "Deep denoising neural network assisted compressive channel estimation for mmwave intelligent reflecting surfaces," *IEEE Trans. Veh. Tech.*, vol. 69, no. 8, pp. 9223–9228, Aug. 2020.

[45] E. Balevi, A. Doshi, and J. G. Andrews, "Massive MIMO channel estimation with an untrained deep neural network," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, pp. 2079–2090, Mar. 2020.