

# Semantic Meta-Split Learning: A TinyML Scheme for Few-Shot Wireless Image Classification

ESLAM ELDEEB<sup>1</sup>, MOHAMMAD SHEHAB<sup>2</sup>, HIRLEY ALVES<sup>1</sup> (Member, IEEE),  
AND MOHAMED-SLIM ALOUINI<sup>2</sup>

<sup>1</sup>Centre for Wireless Communications (CWC), University of Oulu, 90570 Oulu, Finland

<sup>2</sup>CEMSE Division, King Abdullah University of Science and Technology (KAUST), Thuwal 23955, Saudi Arabia

CORRESPONDING AUTHOR: E. ELDEEB (eslam.eldeeb@oulu.fi)

The work of Eslam Eldeeb and Hirley Alves was supported in part by the Research Council of Finland (former Academy of Finland) 6G Flagship Program under Grant 346208 and in part by European Commission through Hexa-X-II under Grant 101095759.

**ABSTRACT** Semantic and goal-oriented (SGO) communication is an emerging technology that only transmits significant information for a given task. Semantic communication encounters many challenges, such as computational complexity at end users, availability of data, and privacy-preserving. This work presents a TinyML-based semantic communication framework for few-shot wireless image classification that integrates split-learning and meta-learning. We exploit split-learning to limit the computations performed by the end-users while ensuring privacy-preserving. In addition, meta-learning overcomes data availability concerns and speeds up training by utilizing similarly trained tasks. The proposed algorithm is tested using a data set of images of hand-written letters. In addition, we present an uncertainty analysis of the predictions using conformal prediction (CP) techniques. Simulation results show that the proposed Semantic-MSL outperforms conventional schemes by achieving a 20% gain in classification accuracy using fewer data points yet less training energy consumption.

**INDEX TERMS** Conformal prediction, meta-learning, semantic communications, split-learning, goal-oriented communication.

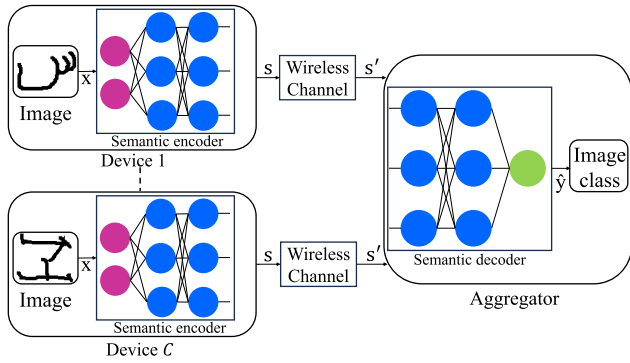
## I. INTRODUCTION

THE road to intelligent communication systems encompasses a paradigm shift from conventional communication models to SGO communication frameworks [1]. SGO communication involves the transmission of only the essential semantics of a message instead of the entire message, which helps to conserve power and improve the spectral efficiency of the communication system [2]. Semantic communication has shown great promise in a wide range of applications. For instance, it has been used for efficient wireless image transmission and classification [3], natural language processing [4], speech processing [5], and multi-modal fusion [6].

Semantic communication enables the receiver to handle various tasks by using only the relevant data that is useful for each specific task. This approach results in more efficient data compression, less signaling overhead and energy

consumption, and higher spectral efficiency for the network. However, current GO communication models faces many challenges related to data collection [7], [8], privacy preserving [9], and amount of computations at the end users [10]. For instance, conventional deep neural network (DNN) models rely on large amounts of data for training, which might not be feasible in many applications due to the high cost of data collection, the scarcity of resources available for data transmission or privacy concerns [11]. In that case, the model usually suffers from high uncertainties due to insufficient data required to build, train, and validate the model.

Meanwhile, semantic communication approaches rely on heavy and power-hungry machine learning (ML) and DNN models for feature extraction and handling large volumes of data. In addition, various use-cases have privacy restrictions on data transmission [12], whereas other applications have low-power / low-resources at end users that make it



**FIGURE 1. Multiple devices aim to classify the correct letters in images sampled from different languages. Each device transmits the semantics of the image through a wireless channel to the aggregator, which predicts the image class and transmits it back to the device.**

challenging to perform complex deep learning computations [13], [14]. To this end, TinyML aims at inventing light yet accurate and communication-efficient ML schemes that consume fewer energy resources to achieve the learning goals [15], [16].

This paper presents an energy and communication-efficient TinyML scheme for wireless image transmission classification, termed Semantic Meta-Split Learning or simply *Semantic-MSL*. The proposed method integrates split-learning [17] and meta-learning [18] to perform few-shot over-the-air (OTA) image classification while ensuring efficient data compression, low computation complexity, and energy consumption at the end users as well as privacy-preservation. To evaluate the performance of the proposed model, we test the algorithm using a data set of images of hand-written letters used for classification.

### A. SPLIT-LEARNING AND META-LEARNING

*Split-learning* [17] is a family of collaborative and distributed ML algorithms, where the learning model, i.e., neural network, is divided into *device-side model* and *aggregator-side model* separated by the *cut-layer* as shown in Fig. 1. The training and testing of the model are executed among the devices and the aggregator. During forward propagation, the devices transmit only the output of their last layer, i.e., *smashed data*, to the aggregator, which, in turn, transmits back the *smashed data's gradients* during back-propagation [19].<sup>1</sup>

The major advantage of split learning over federated learning is that it reduces the heavy computations performed at the device side in federated learning and the high signaling overhead resulting from transmitting the whole model to the aggregator. In split learning, devices and aggregators only exchange *smashed data* and gradients [20], which saves communication energy as required by TinyML

<sup>1</sup>Note that the transmission of gradients is done through a realistic communication channel which includes noise and fading.

techniques. Moreover, split learning provides high levels of privacy-preserving as the data is not shared between the devices and the aggregator, which can access only a portion of the model.

In addition to split-learning, the proposed framework exploits *meta-learning* [18], which is known as *learning-to-learn*. Conventional deep learning requires the model's training from scratch for each new configuration adjusted in the network. In contrast, meta-learning is motivated by utilizing different configurations (tasks) to infer a model that performs well in new configurations. In this context, *model-agnostic meta-learning (MAML)* is a famous family of meta-learning that meta-trains multiple tasks searching for suitable initialization for the weights of the trainable model so that the model converges in a few gradient steps using a few shots of data points. Meta-learning has shown promising performance in different wireless communication use cases, such as channel estimation, symbol demodulation, and modulation classification [21], [22].

### B. RELATED WORK

The advances in semantic communication have been addressed extensively in many recent works. Specifically, the work in [23] proposes a framework that enables the transmission of detected traffic signs from one autonomous vehicle to another. The authors in [24] propose a convolutional neural network (CNN) architecture for wireless image transfer while ensuring security awareness. They focus on privacy preservation without tackling the problem of data availability or energy consumption. In [25], Xie et al. propose transformers-based models for multiple task-oriented communication, whereas the work in [26] discusses resource allocation techniques for semantic communication. In contrast to our work, the proposed methods lack accumulating experience from various tasks (i.e., meta-learning) for training. The work in [27] designs a joint communication and computational probabilistic semantic algorithm for multi-user resource allocation problems, while the authors in [28] investigate a joint communication and computational probabilistic semantic algorithm for distributed reconfigurable intelligent surfaces (RISs)-assisted networks.

Recently, split-learning has also gained popularity in the field of wireless communication. For instance, in the article [29], the authors have introduced a split-learning scheme where learning is carried out across various devices in a parallel manner. However, they have only optimized the cut layer position without considering the uncertainty of different model configurations. The authors of [30] propose a MIMO-based OTA split learning approach that interestingly also estimates the channel efficiency through forward and backward propagation processes, improving the overall system efficiency. Meanwhile, in [31], split and federated learning is utilized for autonomous aerial vehicle (AAV) applications. In [32], federated learning has been combined with split learning to reduce communication latency and improve accuracy within heterogeneous devices.

Meta-learning has gained further attention in wireless communication applications. The work in [33] exploits meta-learning with conformal prediction (CP) to design a well-calibrated model via a few shots of training data. In [34], Zhang et al. combine meta-learning and federated learning for wireless traffic prediction. The authors in [35] optimize the trajectories of multiple AAVs using meta-learning, whereas the authors in [36] optimize the AAV trajectory using meta-reinforcement learning. In addition, meta-learning has shown outstanding performance in other domains, such as intelligent MIMO. The work in [37] proposes an algorithm that combines meta-learning with deep neural networks for the MIMO detector design problem. The proposed framework achieves near-maximum-likelihood performance utilizing meta-learning to outperform existing techniques. To our knowledge, this is the first work to combine split-learning and meta-learning for goal-oriented semantic communications and few-shot wireless image classification.

### C. CONTRIBUTION

This paper introduces a novel TinyML-based semantic communication framework for efficient wireless image transmission and classification. Unlike previous works in the same research direction, the proposed method combines meta-learning and split-learning to provide fast convergence with low complexity and energy consumption. The main contributions of the paper are summarized as follows:

- We propose the Semantic-MSL scheme architecture using convolutional neural networks (CNNs). Split learning ensures low computations on the device side, and meta-learning ensures fast learning using a small amount of data.
- To showcase the proposed framework, we assume a few-shot classification (i.e., only a few images are available for training) of hand-written letters being transmitted between devices and an aggregator.
- We exploit conformal prediction to quantify the uncertainty associated with the proposed model. Simulation results show that the proposed Semantic-MSL framework significantly outperforms SL and conventional deep learning regarding classification accuracy, conformal prediction uncertainty, and energy consumption.
- We show the effect of changing the position of the cut layer on the required computational energy at the users, the deployment time of the algorithm, and the amount of information transmitted.

The proposed scheme shows strong potential for deployment on tiny devices characterized by limited hardware and low energy. Furthermore, it could benefit edge learning frameworks in B5G by splitting the learning process between the client and the server to maintain privacy and save computation resources at the client device [38].

The rest of the paper is organized as follows: Section III describes the system model and defines the problem.

TABLE 1. Important abbreviations and symbols.

Abbreviations	Definition
CNN	Convolutional neural network
CP	Conformal Prediction
MAML	Model agnostic meta-learning
MSL	Meta-split learning
NC	Nonconformity
OTA	over the air
SGD	Stochastic gradient descent
SGO	Semantic and Goal Oriented
TinyML	Tiny Machine Learning

Symbol	Description
$\mathcal{C}$	Set of users
$\text{cov}(\cdot)$	Coverage function
$M$	number of images in each class
$\mathbf{s}$	Transmitted semantic message / smashed data
$\mathbf{s}'$	Received semantic message
$\mathcal{T}$	Set of tasks
$\mathbf{W}^{\mathcal{C}}$	Parameter vector of the device
$\mathbf{W}^{\mathcal{S}}$	Parameter vector of the aggregator
$\mathcal{Y}$	Set of image classes
$\mathbf{y}$	True image class
$\hat{\mathbf{y}}$	Predicted image class
$\beta$	Meta-learning rate
$\eta$	Semantic learning rate
$\Gamma$	Prediction set
$\nabla$	Gradient operator

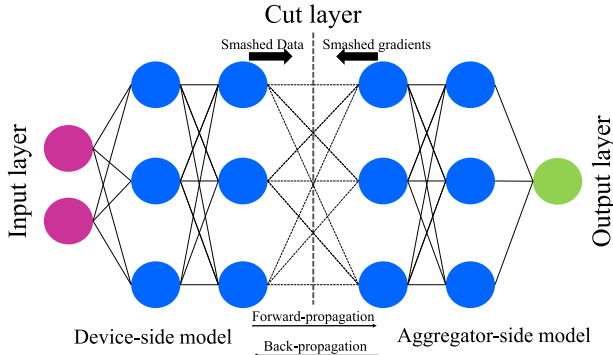
Section II introduces split-learning and meta-learning in brief. In section IV, we present the proposed Semantic-MSL communication framework. Section V explains the performance evaluation metrics, including conformal prediction, whereas section VI shows the experimental results. Section VII concludes the paper. The main symbols and acronyms used in this paper are summarized in TABLE 1.

## II. BACKGROUND

This section presents an overview of split learning and meta-learning. This discussion is necessary to prepare for the proposed meta-split learning-based semantic communication system for wireless transmission and few-shot classification of images.

### A. SPLIT-LEARNING

*Split-learning* [17] is a distributed machine learning framework that divides the machine learning architecture into multiple sectors, i.e., a neural network. These sectors are distributed among multiple devices and aggregators. As shown in Fig. 2, the sector at the device is referred to as *device-side model* with  $\mathbf{W}^{\mathcal{C}}$  parameters vector, whereas the sector at the aggregator is called *aggregator-side model* with  $\mathbf{W}^{\mathcal{S}}$  parameters vector. The layer dividing the device-side and aggregator-side models is called the *cut layer*. Consider a model whose input  $\mathbf{x}$  corresponds to a target output  $\mathbf{y}$ . Herein, the input layer exists at the device side, where the input is forwarded through the device-side model until the cut layer. The output of the device-side model is called *smashed data*,  $\mathbf{s}$ ,



**FIGURE 2.** Illustration of the architecture of split learning. The model is divided at the cut layer into a device-side model and an aggregator-side model, where the training is taking part at both entities.

which is transmitted to the aggregator-side model

$$\mathbf{s} = \mathbf{W}^C \mathbf{x}. \quad (1)$$

Ideally, the smashed data is received at the aggregator side and considered the input to the aggregator-side model. The smashed data propagates through the aggregator-side model, giving

$$\hat{\mathbf{y}} = \mathbf{W}^S \mathbf{s}, \quad (2)$$

where  $\hat{\mathbf{y}}$  is the output of the aggregator-side model. Without loss of generality, the output  $\hat{\mathbf{y}}$  is compared to the target output  $\mathbf{y}$  using an error measure function, i.e., an arbitrary loss function  $\mathcal{L}_{\mathbf{W}^C, \mathbf{W}^S}(\mathbf{y}, \hat{\mathbf{y}})$ . The parameters of the aggregator-side model are updated using stochastic gradient descent (SGD)

$$\mathbf{W}^S \leftarrow \mathbf{W}^S - \eta \nabla_{\mathbf{W}^S} \mathcal{L}_{\mathbf{W}^C, \mathbf{W}^S}(\mathbf{y}, \hat{\mathbf{y}}), \quad (3)$$

where  $\eta$  is the semantic learning rate and  $\nabla_{\mathbf{W}^S} \mathcal{L}_{\mathbf{W}^C, \mathbf{W}^S}(\mathbf{y}, \hat{\mathbf{y}})$  is the gradient calculated concerning the aggregator-side parameters until the cut-layer. Afterward, the smashed data's gradients are transmitted back to the device-side model, where the parameters of the device-side model are updated via

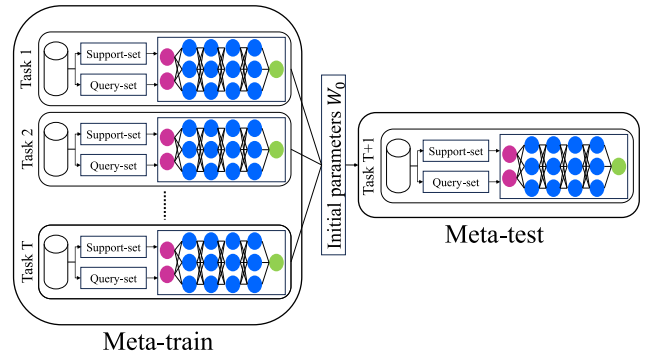
$$\mathbf{W}^C \leftarrow \mathbf{W}^C - \eta \nabla_{\mathbf{W}^C} \mathcal{L}_{\mathbf{W}^C, \mathbf{W}^S}(\mathbf{y}, \hat{\mathbf{y}}), \quad (4)$$

where  $\nabla_{\mathbf{W}^C} \mathcal{L}_{\mathbf{W}^C, \mathbf{W}^S}(\mathbf{y}, \hat{\mathbf{y}})$  is the gradient calculated concerning the device-side parameters (i.e., the way back to the input). This procedure is repeated until convergence.

Split learning reduces the computational load at the devices by dividing the model between the devices and the aggregator [39]. The reduction depends on the position of the cut layer. The earlier the cut layer, the lower the computational load on the device. However, this comes at the cost of a larger size of the transmitted message and lower information security.

## B. META-LEARNING

*Meta-learning* [18], known as *learning-to-learn*, utilizes learning across multiple tasks to infer an optimized learning



**FIGURE 3.** Meta-learning architecture. It consists of i) a meta-train that has a support set, which samples meta-tasks for training, and ii) a query set that tests the meta-learning model on new, unseen tasks.

policy for a new task. As illustrated in Fig. 3, *Model-agnostic meta-learning (MAML)* [18] is a well-known meta-learning algorithm that aims at achieving faster convergence through a few SGD steps by optimizing a common initial parameter  $\mathbf{W}$  across multiple tasks. Assume a task distribution  $p(\tau)$  and  $T$  tasks  $\tau_1, \dots, \tau_T$  sampled randomly and in an i.i.d. fashion from the distribution. These tasks are fed to the *meta-training* phase, where each task  $\tau_i$  comprises a *support-set*, which is used for training, and a *query-test*, which is utilized for testing.

Consider a model with parameter  $\mathbf{W}$  and a loss function  $\mathcal{L}_{\mathbf{W}}(\tau_i)$  adopted to train each task  $\tau_i$

$$\mathbf{W}'_i \leftarrow \mathbf{W} - \eta \nabla_{\mathbf{W}} \mathcal{L}_{\mathbf{W}}(\tau_i), \quad (5)$$

where  $\mathbf{W}'_i$  is the updated parameters for task  $\tau_i$ . The same update is applied to all the sampled tasks, and the meta-loss is calculated using the updated parameters for each task

$$\mathcal{L}_{\text{meta}} = \sum_{i=1}^T \mathcal{L}_{\mathbf{W}'_i}(\tau_i). \quad (6)$$

The meta-optimization is performed collaboratively across all the tasks by using SGD with the meta-loss

$$\mathbf{W} \leftarrow \mathbf{W} - \beta \nabla_{\mathbf{W}} \mathcal{L}_{\text{meta}}, \quad (7)$$

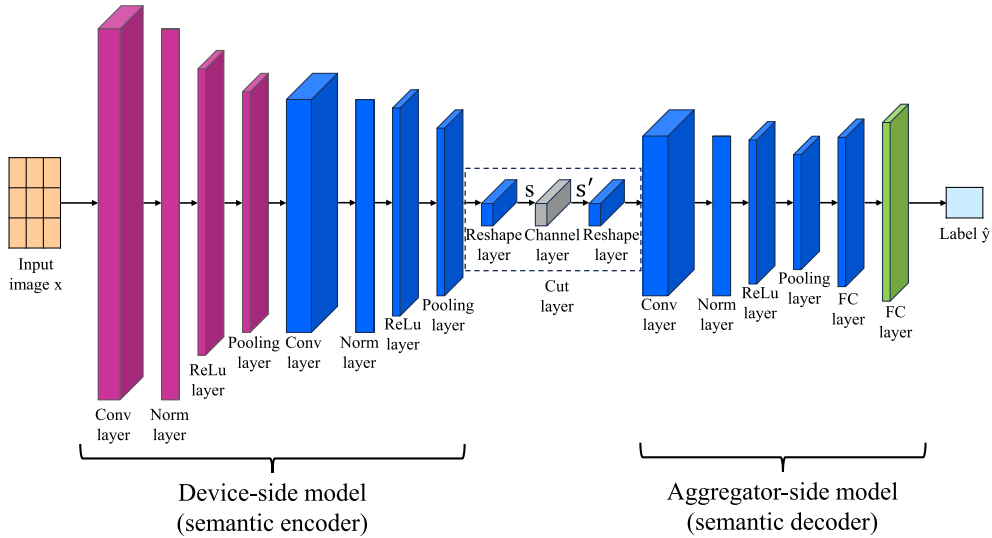
where  $\beta$  is the meta-learning rate.

The *meta-testing* phase adopts a new task  $\tau_{T+1}$  composed of a support set and a query test. The optimized parameters  $\mathbf{W}$  are the initial parameters for the new sampled task, and throughout a few SGD steps on the optimized parameters, the model converges. The number of samples in the support set (in both meta-training and meta-testing) is called *shots*  $K$ , where using a small number of shots is referred to as *few-shot learning*.

## III. SYSTEM MODEL AND PROBLEM FORMULATION

### A. SYSTEM MODEL

As shown in Fig. 1, consider a set of users  $\mathcal{C} = \{1, 2, \dots, C\}$ , where  $C = |\mathcal{C}|$ . Each user transmits an image  $\mathbf{x}$  to an



**FIGURE 4.** The proposed Semantic-MSL architecture. The model consists of multiple convolutional layers, each followed by a normalization layer, a ReLu layer, and a pooling layer. At the end, multiple fully connected layers output the predicted class. At the cut layer, a reshape layer corrects the size of the smashed data to be transmitted through the channel.

aggregator (*e.g.*, BS) through a wireless channel. Consider a task distribution  $p(\tau)$  and  $T$  tasks forming the set  $\mathcal{T} = \{\tau_1, \dots, \tau_T\}$ , where each user  $c$  samples a task from the set of tasks  $\mathcal{T}$ . Each task composes of multiple image classes forming the set  $\mathcal{Y} = \{1, 2, \dots, Y\}$ , where  $\mathbf{Y} = |\mathcal{Y}|$ . Each class contains  $M = |\mathcal{M}|$  images defined as  $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_M\}$ . On the user side, the semantic encoder encodes the image as

$$\mathbf{s} = f_C(\mathbf{x}), \quad (8)$$

where  $\mathbf{s} \in \mathbb{R}^{N \times 1}$  is the transmitted semantic message,  $N$  is the size of the message, and  $f_C(\cdot)$  is the semantic encoder parameterized by the parameters vector  $\mathbf{W}^C$  [40]. The encoded message is modulated using Quadrature amplitude modulation (QAM) modulation and transmitted to the receiver. The transmitted signal has a power constraint  $\frac{1}{N} \mathbb{E}\{\mathbf{x}^2\} \leq p$ .

Consider an additive white Gaussian noise (AWGN) channel with noise distribution  $\mathbf{n} \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$ , where  $\sigma^2$  is the noise power

$$\mathbf{s}' = \mathbf{h} \mathbf{s} + \mathbf{n}, \quad (9)$$

where  $\mathbf{h}$  is the channel component that comprises a Rayleigh flat fading. At the aggregator, the semantic message is decoded through the semantic decoder to depict the class of the image

$$\hat{\mathbf{y}} = f_S(\mathbf{s}'), \quad (10)$$

where  $f_S(\cdot)$  is the semantic decoder parameterized by the parameters vector  $\mathbf{W}^S$ .

## B. PROBLEM DEFINITION

The main objective is to estimate the true class of each image using only the transmitted semantic information of the image.

In addition, we aim to perform this estimation with the lowest possible number of images. In particular, the target is to find the optimum model parameters  $\mathbf{W}^C$  and  $\mathbf{W}^S$  correspond to the best models  $f_C(\cdot)$  and  $f_S(\cdot)$ , respectively, so that the true classes of the images can be estimated using a few shots of training images. This optimization problem is formulated as

$$\mathbf{P1} : \min_{\mathbf{W}^C, \mathbf{W}^S} \sum_C \sum_T \left[ \sum_K \mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) \right], \quad (11a)$$

$$\text{s.t. } K \leq K_{\text{th}}, \quad (11b)$$

$$\hat{\mathbf{y}}, \mathbf{y} \in \mathcal{Y}, \quad (11c)$$

where  $\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}})$  is a loss function that measures the distance between the true class and the predicted class (*e.g.*, cross-entropy loss function), and  $K$  represents the number of images sampled per class from each task. The constraint (11b) forces the number of sampled images per class  $K$  to be lower than or equal to a user-chosen threshold  $K_{\text{th}}$ . In contrast, the constraint (11c) ensures that the output  $\mathbf{y}$  and the estimated output  $\hat{\mathbf{y}}$  are in the set of all possible classes. The variables ( $\mathbf{W}^C$  and  $\mathbf{W}^S$ ) can be modeled and solved using an adequate deep learning technique, such as CNN due to its powerful ability to perform feature extraction on images.

## IV. SEMANTIC META-SPLIT LEARNING FRAMEWORK

This section presents the novel Semantic-MSL learning-based framework for wireless image transmission. This approach combines split learning with meta-learning to formulate a low computational device-side architecture wireless image classification using a small number of images. As shown in Fig. 1, the device feeds the semantic information of an image forward through to the device-side network, *i.e.*,

the semantic encoder. Then, the receiver estimates the image class through the aggregator-side network, i.e., the semantic decoder. In addition, we assume a few-shot classification problem, where only a small number of images are available for each task.

The integration of meta-learning and split-learning ensures both rapid adaptation and computational efficiency. Theoretically, meta-learning optimizes a model’s initialization such that it can generalize across new tasks with minimal updates, significantly reducing the number of required gradient steps compared to conventional learning schemes. Additionally, split-learning reduces the computation load at the device by offloading deep neural network layers to the aggregator, thereby minimizing energy consumption while maintaining privacy. The combination of these two approaches enables a more resource-efficient and adaptive learning process, making the framework well-suited for TinyML applications with constrained data and power budgets.

To this end, for a particular task  $\tau_i$ , the semantic decoder with initial parameters  $\mathbf{W}^S$  and a loss function  $\mathcal{L}_{\mathbf{W}^C, \mathbf{W}^S}(\mathbf{y}, \hat{\mathbf{y}}; \tau_i)$  is trained at the aggregator-side as

$$\mathbf{W}'_i{}^S \leftarrow \mathbf{W}^S - \eta \nabla_{\mathbf{W}^S} \mathcal{L}_{\mathbf{W}^C, \mathbf{W}^S}(\mathbf{y}, \hat{\mathbf{y}}; \tau_i), \quad (12)$$

where  $\mathbf{W}'_i{}^S$  is the task-specific updated semantic decoder parameters. Similarly, the semantic encoder with initial parameter  $\mathbf{W}^C$  would be trained at the device-side as

$$\mathbf{W}'_i{}^C \leftarrow \mathbf{W}^C - \eta \nabla_{\mathbf{W}^C} \mathcal{L}_{\mathbf{W}^C, \mathbf{W}^S}(\mathbf{y}, \hat{\mathbf{y}}; \tau_i), \quad (13)$$

where  $\mathbf{W}'_i{}^C$  is the task-specific updated semantic encoder parameters and the gradient  $\nabla_{\mathbf{W}^C} \mathcal{L}_{\mathbf{W}^C, \mathbf{W}^S}(\mathbf{y}, \hat{\mathbf{y}}; \tau_i)$  is transmitted from the aggregator to the device through the wireless channel. Afterward, the newly updated parameters  $\mathbf{W}'_i{}^C$  and  $\mathbf{W}'_i{}^S$  are used to estimate the meta-split loss across the tasks used for meta-split training

$$\mathcal{L}_{\text{meta-split}} = \sum_{i=1}^T \mathcal{L}_{\mathbf{W}'_i{}^C, \mathbf{W}'_i{}^S}(\mathbf{y}, \hat{\mathbf{y}}; \tau_i). \quad (14)$$

Finally, the initial parameters are updated using the meta-split update

$$\mathbf{W}^C \leftarrow \mathbf{W}^C - \beta \nabla_{\mathbf{W}^C} \mathcal{L}_{\text{meta-split}}, \quad (15)$$

$$\mathbf{W}^S \leftarrow \mathbf{W}^S - \beta \nabla_{\mathbf{W}^S} \mathcal{L}_{\text{meta-split}}, \quad (16)$$

where  $\beta$  is the meta-learning rate. The aforementioned steps are repeated till convergence. During testing, a few SDG steps are carried out on the semantic encoder and semantic decoder with a few shots ( $K$ ) on a new task  $\tau_{T+1}$ .

The proposed semantic-MSL framework addresses two key challenges in wireless image transmission; computational efficiency and adaptability to new tasks. Split learning reduces the computational burden on device-side hardware by offloading complex processing tasks to the aggregator-side network. At the same time, meta-learning enables the model to adapt to unseen tasks with minimal data quickly. Moreover, the gradient-sharing mechanism ensures privacy and security by transmitting only the learned representations rather

than raw data. These capabilities are particularly beneficial in scenarios that include limited computational resources, privacy-sensitive applications, or scenarios with limited data available.

Fig. 4 illustrates the proposed *convolutional neural network (CNN)* for both semantic encoder and semantic decoder. The model consists of multiple convolutional layers, each followed by a normalization layer, a ReLu layer, and a pooling (max) layer. After the convolutional layers, multiple fully-connected (FC) layers are followed by a softmax layer that outputs the estimated label (class). The cut layer controls the semantic encoder and the semantic decoder. The convolution layers are feature extraction layers, and thus, the later the cut layer, the lower the dimension of the transmitted semantic message  $\mathbf{s}$ . In contrast, the earlier the cut layer, the lower the computations performed on the device side. This illustrates the trade-off between the rate of encoding and the computational complexity.

In practice, determining the optimal cut layer is a crucial design consideration for achieving the best trade-off between communication/computation requirements and semantic communication performance (accuracy of achieving the objective). In addition, the framework accounts for wireless channel conditions, ensuring that the transmitted semantic message remains robust to noise and interference. Adjusting the cut layer based on real-time conditions, such as channel quality or device battery level, supports a robust algorithmic design. Furthermore, by controlling the cut layer position, the number of trainable parameters at the device side is significantly reduced, lowering the memory footprint and making the model feasible for TinyML deployment. This balance between computational efficiency, space complexity, and adaptation speed makes the framework well-suited for constrained wireless environments. Algorithm 1 summarizes the proposed Semantic-MSL framework. In the next section, we present the key performance metrics needed to evaluate the proposed algorithm in terms of communication/computation requirements, image classification performance, effect of channel conditions, and model uncertainties.

## V. KEY PERFORMANCE METRICS

This section presents the key performance metrics leveraged to evaluate the proposed model compared to other baseline schemes. For instance, classification accuracy is defined as the total number of true classified images divided by the total number of images. The precision, recall, and f1-score are considered the fundamental metric in evaluating classification problems [41]. However, these metrics lack measuring the degree of uncertainty and the model’s trustworthiness, which is usually common in deep learning models with limited access to data points [42].

*Conformal prediction (CP)* [43] is a framework that quantifies the degree of uncertainty in predictions of predictive models (such as neural networks). CP calibrates the models by generating prediction sets instead of a single prediction.

**Algorithm 1** The Proposed Semantic-MSL Algorithm

```

1 Input: Semantic learning rate  $\eta$ , meta-learning rate  $\beta$ ,
   number of tasks  $T$ , and number of training epochs  $E$ 
2 Output: Initial parameters  $\mathbf{W}^C$  and  $\mathbf{W}^S$ 
3 Initialize network parameters  $\mathbf{W}^C$  and  $\mathbf{W}^S$ 
4 for epochs  $e$  in  $\{1, \dots, E\}$  do
5     Sample  $T$  tasks from the distribution  $p(\tau)$ 
6     for each sampled task in  $\{\tau_1, \dots, \tau_T\}$  do
7         Sample  $K$  shots for each class in each task
8         Update the model parameters  $\mathbf{W}^C$  and  $\mathbf{W}^S$ 
           using (12) and (13)
9     end
10    Calculate the meta-split loss  $\mathcal{L}_{\text{meta-split}}$  using (14)
11    Update the initial parameters  $\mathbf{W}^C$  and  $\mathbf{W}^S$  using (16)
       and (15)
12 end
13 Return model initial parameters  $\mathbf{W}^C$  and  $\mathbf{W}^S$ 
    
```

Consider an input  $\mathbf{x}$  corresponding to an output  $\mathbf{y}$  from the set of outputs  $\mathcal{Y}$ . Relying on the fact that the training and testing of input data are exchangeable, CP produces a subset of the output set, i.e., *prediction set*  $\Gamma$ , which contains the true output with a probability  $1 - \alpha$ , where  $\alpha \in [0, 1]$  [43].

CP's prediction set is evaluated in terms of *coverage*  $\text{cov}(\Gamma)$  and *inefficiency*  $\text{ineff}(\Gamma)$ . The former measures the probability of the true label in the prediction set, whereas the latter is the size of the prediction set. The coverage and the inefficiency are formulated as

$$\text{cov}(\Gamma) = P(\mathbf{y} \in \Gamma), \quad (17)$$

$$\text{ineff}(\Gamma) = \mathbb{E} [|\Gamma|], \quad (18)$$

where the average  $\mathbb{E} [\cdot]$  is taken over the given data points. A model that generates a prediction set that is equivalent to the entire output set yields the maximum possible coverage  $\text{cov}(\Gamma) = 1$  at the cost of having the worst inefficiency  $\text{ineff}(\Gamma) = |\mathcal{Y}|$ . In contrast, a model that generates an empty prediction set achieves the best possible inefficiency  $\text{ineff}(\Gamma) = 0$  at the cost of the coverage  $\text{cov}(\Gamma) = 0$ . Hence, a well-calibrated model would maintain the trade-off between coverage and inefficiency by achieving no lower than  $1 - \alpha$  coverage with a relatively low inefficiency.

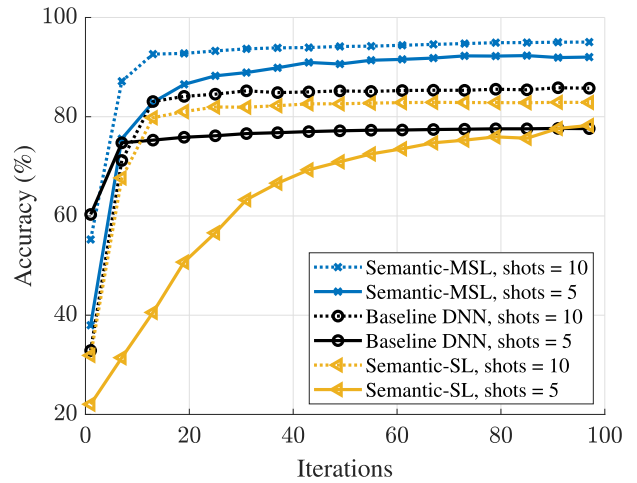
*Validation-based CP (VB-CP)* is a class of CP in which the data points are divided into calibration data set  $D^c$  and validation data  $D^v$ . Using the calibration data, we calculate the *nonconformity* (NC) score, which is a function that describes how poor an output is for a given input. For instance, a well-known NC score is

$$\text{NC}(D^c) = 1 - \text{Softmax}(D^c). \quad (19)$$

Then, we compute the  $q^{\text{th}}$  quantile  $\hat{q} = \frac{[(n+1)(1-\alpha)]}{n}$  of the NC scores of the calibration data, where  $n$  is the size of the calibration data. For the validation data, the prediction set is generated, such that it includes all the classes that have

**TABLE 2.** Simulation parameters and hyperparameters.

Parameter	Value	Parameter	Value
$T$	20	$Y$	10
$M$	20	$K_{\text{th}}$	5
$\alpha$	0.1	epochs $E$	1000
Conv. layers	3 layers (64 neurons)	FC. layers	2 layers (64 neurons)
Kernel size	3	stride	2
$\eta$	0.001	$\beta$	0.01
Optimizer	Adam	loss	CrossEntropy


**FIGURE 5.** The convergence of the accuracy as a function of the SGD steps (iterations) of the proposed Semantic-MSL compared to the Semantic-SL scheme.

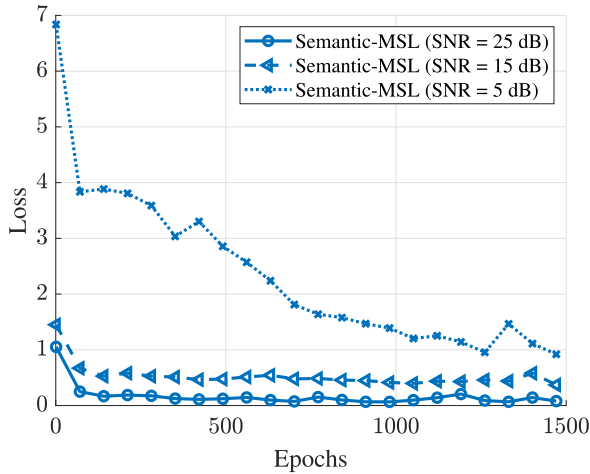
NC-scores smaller than or equal to the empirical quantile [44]

$$\Gamma = \{\mathbf{y}' \in \mathcal{Y} \mid \text{NC}(\mathbf{x}, \mathbf{y}'; D^v) \leq q^{\text{th}} \text{NC}(D^c)\}. \quad (20)$$

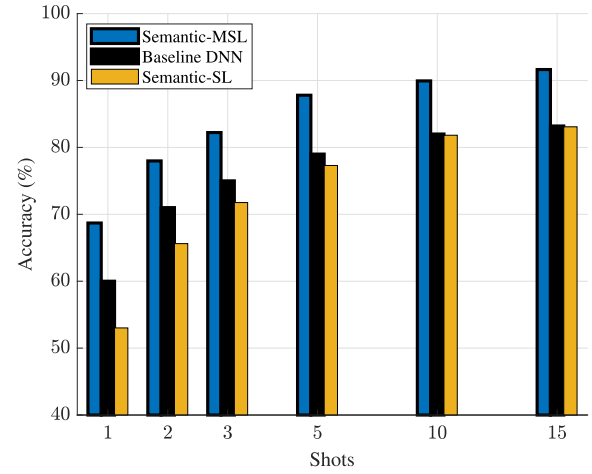
## VI. NUMERICAL RESULTS

This section presents the simulation results of the proposed Semantic-MSL framework. We use 3 convolution neural network with 64 neurons each followed by two fully connected neural networks with two hidden layers of size 64 and ReLU activation functions. The experiments are implemented using Pytorch on a single NVIDIA Tesla V100 GPU. We implement the experiments using Omniglot data set [45], which consists of 1623 classes of letters from 50 different languages. Each class contains 20 different hand-written images. Table 2 illustrates the parameters used in the simulation. The proposed Semantic-MSL model is compared to Semantic-split learning (Semantic-SL), which only uses split learning, and a baseline DNN model, where the whole model is built and trained at the device. Moreover, we test the proposed model while adjusting the number of training tasks, number of shots, and position of the cut layer.

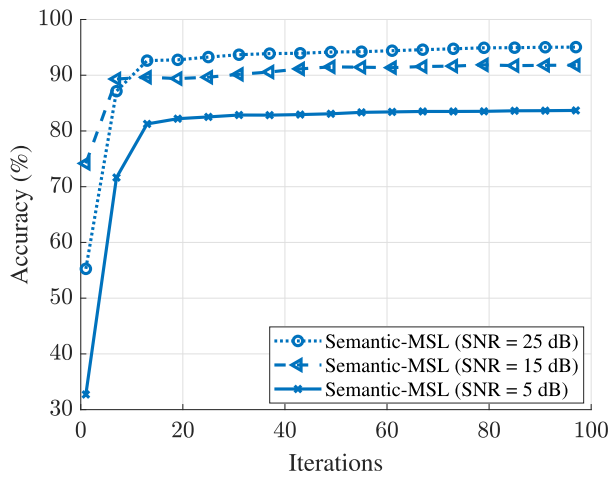
In Fig. 6, we illustrate the impact of channel conditions, specifically the signal-to-noise ratio (SNR), on the convergence behavior of the proposed Semantic-MSL algorithm and its classification accuracy. First, in Fig. 6a, When the SNR is as low as 5 dB, the model consumes the whole training epochs



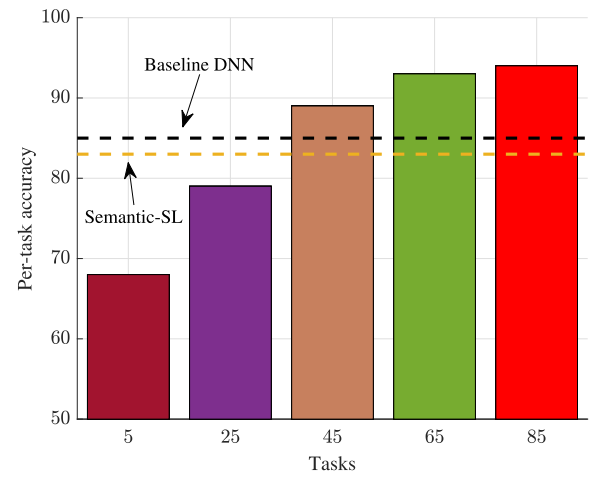
(a) Loss for different SNR values



(a) Accuracy for different number of shots



(b) Accuracy for different SNR values



(b) Accuracy for different number of tasks

**FIGURE 6. A report of loss values and classification accuracy. In (a), loss values are plotted as function of meta-training epochs. In (b), classification accuracy is reported as function of meta-testing iterations.**

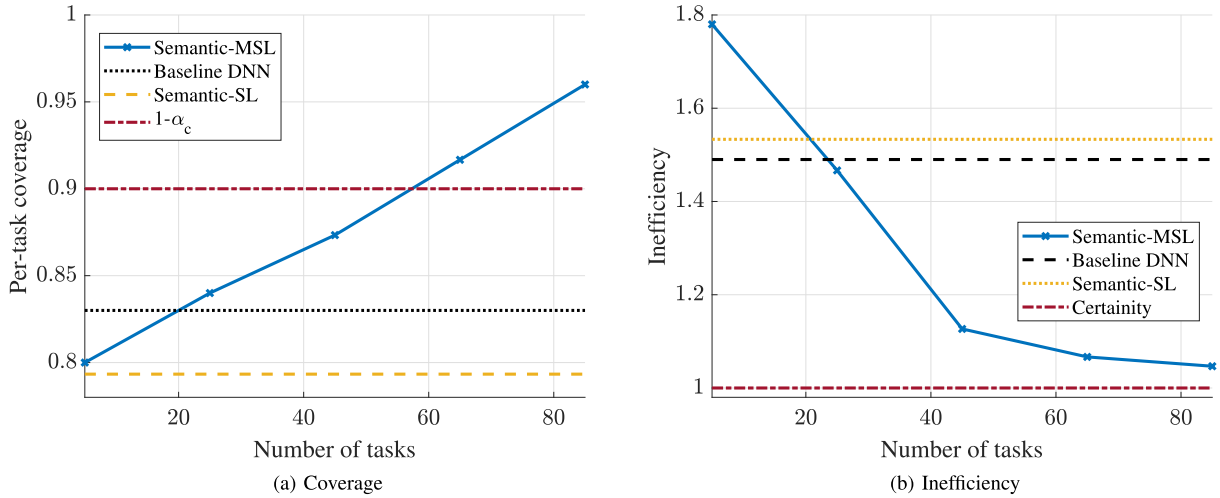
without achieving convergence. In contrast, for SNR values above 15 dB, the model exhibits rapid loss reduction within only a few training epochs, indicating efficient convergence. Additionally, Fig. 6b demonstrates that under poor channel conditions (SNR = 5 dB, the accuracy is upper bounded at approximately 82%. In contrast, when the channel conditions improve (SNR = 15 dB), the model achieves substantially higher classification accuracy of 95%. This highlights the impact of poor channel conditions that can lead to serious problems, such as packet loss and longer latency, on the system performance.

In Fig. 5, we report the achievable classification accuracy after a given number of iterations of SGD steps using 5-shots of images from each class. In this experiment, Semantic-MSL is trained on 65 different tasks and compared to the baseline DNN and Semantic-SL schemes. We can notice that the Semantic-MSL converges to an optimum accuracy of 95% after only 20 SGD steps outperforming both DNN

**FIGURE 7. The classification accuracy is a function of both the number of shots (number of images used in training) and the number of meta-tasks. In (a), compares the accuracy of Semantic-MSL, Semantic-SL, DNN schemes as function of the number of shots. In (b), the Semantic-MSL scheme's per-task accuracy is plotted as a function of the number of meta-tasks.**

and SL. This occurs due to the enhanced initialization of the MAML algorithm obtained from meta-training of 65 tasks. In contrast, the Semantic-SL scheme reaches less than 60% accuracy after 20 SGD steps, which is a bit lower than the DNN, despite SL's superiority over DNN in terms of reduced power consumption. In addition, it spends 100 SGD steps to only reach a sub-optimum accuracy of 82%. This highlights the benefits of meta-learning in utilizing experience from other tasks to improve and speed up learning new tasks.

Fig. 7 depicts the effect of the number of image shots used in training for each class in a particular task on the classification accuracy for both Semantic-MSL and Semantic-SL schemes. In addition, we show the effect of the number of meta-training tasks on the classification accuracy for the proposed Semantic-MSL scheme. Herein, the number of training iterations is fixed to 30 SGD steps. In Fig. 7a, we observe that



**FIGURE 8.** Coverage and inefficiency of the Semantic-MSL and Semantic-SL schemes across different tasks.

increasing the number of training shots improves the classification accuracy. Interestingly, the proposed Semantic-MSL scheme achieves more than 91% accuracy with only 5 shots (we keep the number of meta-tasks fixed to 45 tasks). In contrast, Semantic-SL and DNN fail to exceed 83% accuracy even with 15 shots. Moreover, Fig. 7b illustrates that increasing the number of meta-training tasks enhances accuracy. The proposed model requires less than 45 training tasks to outperform the baseline schemes, Baseline DNN, and Semantic-SL. We can notice that classification accuracy does not improve much when the number of tasks exceeds 65. Increasing the number of tasks over 85 does not provide any noticeable gain in the classification accuracy as saturation behavior is noticed.

The next experiment investigates the uncertainty of the proposed Semantic-MSL model compared to Semantic-SL and DNN using the discussed conformal prediction metrics as depicted in Fig. 8. The first aspect of conformal prediction metrics is the coverage, which measures the probability of the true class in the prediction set  $\Gamma$ , shown in Fig. 8a. Setting  $\alpha = 0.1$ , we observe that increasing the number of tasks enhances Semantic-MSL coverage as it approaches 1 for a larger number of meta-tasks. In contrast, the coverage of Semantic-SL and DNN is always constant and inferior to Semantic-MSL. To obtain coverage larger than or equal to  $1 - \alpha$ , i.e., 0.9, meta-learning requires 55 tasks or more. In Fig. 8b, we observe the inefficiency, which is the average size of the prediction set  $\Gamma$ . Semantic-SL has a constant set size of 1.53 compared to Semantic-MSL, which is affected by the number of tasks. The higher the number of meta-tasks, the closer the model to predict certainly (i.e., approaches inefficiency of 1).

To elaborate further on the computational complexity of the proposed model, we utilize ECO2AI<sup>2</sup> tool to estimate

<sup>2</sup>ECO2AI [46] is an open-source python library that tackles the CPU and GPU consumption and estimates the equivalent CO2 emissions. It aims to achieve models with low computational costs.

**TABLE 3.** Semantic-MSL training duration, Energy consumption, CO2 Emissions, and the coverage accuracy for different meta-tasks.

Tasks	Duration (s)	Energy (Wh)	CO2 emissions (g)	Coverage
5	336.72	7.77	1.10	0.800
25	1659.12	35.48	5.84	0.840
45	2892.28	72.28	10.25	0.873
65	3819.81	96.45	13.68	0.917
85	6041.63	217.97	30.92	0.960

the computational costs using different training tasks and different split layer configurations. Table 3<sup>3</sup> reports the time complexity, computational energy consumption, and the accuracy coverage of the proposed Semantic-MSL scheme while adjusting the number of tasks in the MAML algorithm. We notice that the training duration, energy consumption, and CO2 footprint increase as meta-tasks increase. Similarly, the coverage accuracy is enhanced by increasing the number of meta-tasks. To this end, setting the number of tasks to [45–65] renders relatively high accuracy and coverage while being conservative over the training duration and the required computational energy.

In Table 4, we report the computation energy, the size of the transmitted semantic message, the communication energy, and the deployment time (inference time) of the device-side model while adjusting the position of the cut-layer. The baseline DNN has the highest computation energy and inference time. Setting the cut layer after the third convolutional layer (3-Conv) leaves the aggregator-side model with only the fully connected layers. In that case, the size of the transmitted message is small (0.248 KB) since the convolutional layers are

<sup>3</sup>The duration corresponds to the total wall-clock time required to train the model using a single NVIDIA Tesla V100 GPU. Energy consumption is computed using the ECO2AI library, which monitors CPU/GPU utilization, memory usage, and real-time frequency to estimate energy consumption and associated CO2 emissions. For details, see ECO2AI [46].

**TABLE 4. Reporting the computation and communication energy consumption during training, as well as the size of the transmitted data and device inference time during testing the semantic-MSL model while adjusting the position of the cut layer compared to the baseline model.**

Model / Cut-layer	Computation energy (Wh)	Communication energy (Wh)	Size (KB)	Inference time (ms)
Baseline	44.66	–	–	889.54
3-Conv	41.20	$3.45 \times 10^{-3}$	0.248	770.54
2-Conv	35.48	$4.51 \times 10^{-2}$	6.200	479.25
1-Conv	29.33	$7.83 \times 10^{-1}$	14.912	287.64

feature extractors, and the communication energy is relatively small compared to other cut-layer positions. However, it loads the device-side model with higher computational energy and a larger deployment time. In contrast, shifting the cut-layer position to be after the second convolutional layer (2-Conv) or the first convolutional layer (1-Conv) reduces the required computational energy and deployment time at the devices at the costs of increasing the size of the transmitted message and the communication energy. This highlights the trade-off between computational energy, deployment time, communication energy, and overhead. Note that the computation energy is dominant over the communication energy in all cases; therefore, reducing the computation energy by moving the cut layer closer to the device size (i.e., 1-Conv) reduces the overall energy consumption of the device. However, this comes at the cost of higher communication overhead and the risk of privacy incursions, which is an open area for future research.

To better illustrate the impact of the cut-layer position on device-side complexity, we evaluate the space complexity in terms of the number of trainable parameters. The number of parameters in a convolutional layer is given by [47]

$$\#parameters(Conv) = ((n_k^2 \cdot c_{in}) + 1) \cdot c_{out}, \quad (21)$$

where  $n_k$  represents the kernel size,  $c_{in}$  denotes the number of input and  $c_{out}$  denotes the number of output channels. Similarly, the number of parameters in a fully connected (FC) layer is calculated as

$$\#parameters(FC) = (n_{in} \cdot n_{out}) + n_{out}, \quad (22)$$

where  $n_{in}$  refers to the size of the input of the hidden layer and  $n_{out}$  refers to the size of the output of the hidden layer.

Using (21) and (22), we compute the number of trainable parameters for the for the deployed CNN architecture detailed in Table 2. The three convolution layers contain 640, 36928 and 36928 trainable parameters, respectively, while the FC layers contribute a total of 20933 parameters. Setting the cut layer after the third convolutional layer (3-Conv) results in 74496 trainable parameters at the device, while the aggregator learns only the FC layers. Moving the cut layer after the

second convolutional layer (2-conv) reduces the device-side parameters to 37568 leaving 57861 for the aggregator. In contrast, placing the cut-layer after the first convolutional layer (1-conv) minimizes the device-side parameters to just 640, but increases the aggregator’s parameter load to 94789. This highlights the flexibility of split-learning in balancing computational load, making it a powerful tool for TinyML applications.

## VII. CONCLUSION

In this paper, we proposed Semantic-MSL as a novel Tiny-ML framework for few-shot wireless image classification. The devices transmit the semantics of an image over a wireless channel to the aggregator, which predicts the true class of the image via meta-training over several classification tasks. Numerical results depicted the benefits of combining meta-learning with split-learning, which achieves 20% higher classification accuracy than using only split-learning while using fewer training shots simultaneously. Regarding CP aspects, Semantic-MSL renders high coverage and low inefficiency. Moreover, the overall energy consumption of the proposed Semantic-MSL is significantly reduced by moving the cut layer closer to the device side.

In summary, Semantic-MSL outperforms baseline DNN and Semantic-SL regarding accuracy, CP accuracy, CP inefficiency, and energy consumption. The proposed framework can be further adopted in tasks other than wireless image classification, such as image reconstruction and reinforcement learning. Exploiting meta-conformal prediction in optimizing the split learning model is left for future research.

## REFERENCES

- [1] C. Zhang, H. Zou, S. Lasaulce, W. Saad, M. Kountouris, and M. Bennis, “Goal-oriented communications for the IoT and application to data compression,” *IEEE Internet Things Mag.*, vol. 5, no. 4, pp. 58–63, Dec. 2022.
- [2] Z. Qin, X. Tao, J. Lu, W. Tong, and G. Y. Li, “Semantic communications: Principles and challenges,” 2021, *arXiv:2201.01389*.
- [3] E. Boursoulatze, D. B. Kurka, and D. Gündüz, “Deep joint source-channel coding for wireless image transmission,” *IEEE Trans. Cogn. Commun. Netw.*, vol. 5, no. 3, pp. 567–579, Sep. 2019.
- [4] Y. Sheng, F. Li, L. Liang, and S. Jin, “A multi-task semantic communication system for natural language processing,” in *Proc. 96th IEEE Veh. Technol. Conf.*, Jul. 2022, pp. 1–5.

- [5] Z. Weng and Z. Qin, "Semantic communication systems for speech transmission," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 8, pp. 2434–2444, Aug. 2021.
- [6] H. Xie, Z. Qin, and G. Y. Li, "Task-oriented multi-user semantic communications for VQA," *IEEE Wireless Commun. Lett.*, vol. 11, no. 3, pp. 553–557, Mar. 2021.
- [7] M. Z. Chowdhury, M. Shahjalal, S. Ahmed, and Y. M. Jang, "6G wireless communication systems: Applications, requirements, technologies, challenges, and research directions," *IEEE Open J. Commun. Soc.*, vol. 1, pp. 957–975, 2020.
- [8] E. Eldeeb, H. Sifaou, O. Simeone, M. Shehab, and H. Alves, "Conservative and risk-aware offline multi-agent reinforcement learning," 2024, *arXiv:2402.08421*.
- [9] X. S. Shen et al., "Data management for future wireless networks: Architecture, privacy preservation, and regulation," *IEEE Netw.*, vol. 35, no. 1, pp. 8–15, Jan. 2021.
- [10] P. Kairouz et al., "Advances and open problems in federated learning," *Found. Trends Mach. Learn.*, vol. 14, nos. 1–2, pp. 1–210, Jun. 2021.
- [11] R. Agarwal, D. Schuurmans, and M. Norouzi, "An optimistic perspective on offline reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, vol. 119, 2020, pp. 104–114. [Online]. Available: <https://proceedings.mlr.press/v119/agarwal20c.html>
- [12] Y. Liang, H. V. Poor, and S. Shamai (Shitz), "Secure communication over fading channels," *IEEE Trans. Inf. Theory*, vol. 54, no. 6, pp. 2470–2492, Jun. 2008.
- [13] A. Singh, P. Vepakomma, O. Gupta, and R. Raskar, "Detailed comparison of communication efficiency of split learning and federated learning," 2019, *arXiv:1909.09145*.
- [14] E. Eldeeb et al., "Traffic prediction and fast uplink for hidden Markov IoT models," *IEEE Internet Things J.*, vol. 9, no. 18, pp. 17172–17184, Sep. 2022.
- [15] H. Han and J. Siebert, "TinyML: A systematic review and synthesis of existing research," in *Proc. Int. Conf. Artif. Intell. Inf. Commun. (ICAIC)*, Feb. 2022, pp. 269–274.
- [16] R. Kallimani, K. Pai, P. Raghuvanshi, S. Iyer, and O. L. A. López, "TinyML: Tools, applications, challenges, and future research directions," *Multimedia Tools Appl.*, vol. 83, no. 10, pp. 29015–29045, Sep. 2023.
- [17] Y. Gao et al., "End-to-end evaluation of federated learning and split learning for Internet of Things," 2020, *arXiv:2003.13376*.
- [18] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, Aug. 2017, pp. 1126–1135. [Online]. Available: <https://proceedings.mlr.press/v70/finn17a.html>
- [19] M. V. da Silva, E. Eldeeb, M. Shehab, H. Alves, and R. D. Souza, "Distributed learning methodologies for massive machine type communication," *IEEE Internet Things Mag.*, vol. 8, no. 1, pp. 102–108, Jan. 2025, doi: [10.1109/IOTM.001.2400093](https://doi.org/10.1109/IOTM.001.2400093).
- [20] W. Wu et al., "Split learning over wireless networks: Parallel design and resource management," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 4, pp. 1051–1066, Apr. 2023.
- [21] H. Issa, M. Shehab, and H. Alves, "Meta-learning based few pilots demodulation and interference cancellation for NOMA uplink," in *Proc. Joint Eur. Conf. Netw. Commun. 6G Summit (EuCNC/6G Summit)*, Jun. 2023, pp. 84–89.
- [22] S. Park, O. Simeone, and J. Kang, "Meta-learning to communicate: Fast end-to-end training for fading channels," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2020, pp. 5075–5079.
- [23] A. D. Raha, M. S. Munir, A. Adhikary, Y. Qiao, S.-B. Park, and C. S. Hong, "An artificial intelligent-driven semantic communication framework for connected autonomous vehicular network," in *Proc. Int. Conf. Inf. Netw. (ICOIN)*, Jan. 2023, pp. 352–357.
- [24] M. Zhang, Y. Li, Z. Zhang, G. Zhu, and C. Zhong, "Wireless image transmission with semantic and security awareness," *IEEE Wireless Commun. Lett.*, vol. 12, no. 8, pp. 1389–1393, Aug. 2023.
- [25] H. Xie, Z. Qin, X. Tao, and K. B. Letaief, "Task-oriented multi-user semantic communications," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 9, pp. 2584–2597, Sep. 2022.
- [26] L. Yan, Z. Qin, R. Zhang, Y. Li, and G. Y. Li, "Resource allocation for text semantic communications," *IEEE Wireless Commun. Lett.*, vol. 11, no. 7, pp. 1394–1398, Jul. 2022.
- [27] Z. Zhao, Z. Yang, M. Chen, Z. Zhang, and H. V. Poor, "A joint communication and computation design for probabilistic semantic communications," *Entropy*, vol. 26, no. 5, p. 394, Apr. 2024. [Online]. Available: <https://www.mdpi.com/1099-4300/26/5/394>
- [28] Z. Zhao et al., "A joint communication and computation design for distributed RIS-assisted probabilistic semantic communication in IIoT," *IEEE Internet Things J.*, vol. 11, no. 16, pp. 26568–26579, Aug. 2024.
- [29] Z. Lin et al., "Efficient parallel split learning over resource-constrained wireless edge networks," *IEEE Trans. Mobile Comput.*, vol. 23, no. 10, pp. 9224–9239, Oct. 2024.
- [30] Y. Yang et al., "Over-the-air split machine learning in wireless MIMO networks," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 4, pp. 1007–1022, Apr. 2023.
- [31] X. Liu, Y. Deng, and T. Mahmoodi, "Wireless distributed learning: A new hybrid split and federated learning approach," *IEEE Trans. Wireless Commun.*, vol. 22, no. 4, pp. 2650–2665, Apr. 2023.
- [32] C. Xu, J. Li, Y. Liu, Y. Ling, and M. Wen, "Accelerating split federated learning over wireless communication networks," *IEEE Trans. Wireless Commun.*, vol. 23, no. 6, pp. 5587–5599, Jun. 2024.
- [33] S. Park, K. M. Cohen, and O. Simeone, "Few-shot calibration of set predictors via meta-learned cross-validation-based conformal prediction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 46, no. 1, pp. 280–291, Jan. 2024.
- [34] L. Zhang, C. Zhang, and B. Shihada, "Efficient wireless traffic prediction at the edge: A federated meta-learning approach," *IEEE Commun. Lett.*, vol. 26, no. 7, pp. 1573–1577, Jul. 2022.
- [35] Y. Hu, M. Chen, W. Saad, H. V. Poor, and S. Cui, "Distributed multi-agent meta learning for trajectory design in wireless drone networks," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 10, pp. 3177–3192, Oct. 2021.
- [36] Y. Hu, M. Chen, W. Saad, H. V. Poor, and S. Cui, "Meta-reinforcement learning for trajectory design in wireless UAV networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2020, pp. 1–6.
- [37] H. Huo, J. Xu, G. Su, W. Xu, and N. Wang, "Intelligent MIMO detection using meta learning," *IEEE Wireless Commun. Lett.*, vol. 11, no. 10, pp. 2205–2209, Oct. 2022.
- [38] W. Xu, Z. Yang, D. W. K. Ng, M. Levorato, Y. C. Eldar, and M. Debbah, "Edge learning for B5G networks with distributed signal processing: Semantic communication, edge computing, and wireless sensing," *IEEE J. Sel. Topics Signal Process.*, vol. 17, no. 1, pp. 9–39, Jan. 2023.
- [39] C. Thapa, M. A. P. Chamikara, and S. A. Camtepe, "Advancements of federated learning towards privacy preservation: From federated learning to split learning," in *Federated Learning Systems: Towards Next-Generation AI*. Springer, 2021, pp. 79–109.
- [40] E. Eldeeb, M. Shehab, and H. Alves, "A multi-task oriented semantic communication framework for autonomous vehicles," 2024, *arXiv:2403.12997*.
- [41] E. Eldeeb, M. Shehab, and H. Alves, "A learning-based fast uplink grant for massive IoT via support vector machines and long short-term memory," *IEEE Internet Things J.*, vol. 9, no. 5, pp. 3889–3898, Mar. 2022.
- [42] S. Bates, A. Angelopoulos, L. Lei, J. Malik, and M. Jordan, "Distribution-free, risk-controlling prediction sets," *J. ACM*, vol. 68, no. 6, pp. 1–34, Dec. 2021.
- [43] A. N. Angelopoulos and S. Bates, "A gentle introduction to conformal prediction and distribution-free uncertainty quantification," 2021, *arXiv:2107.07511*.
- [44] I. Gibbs and E. Candes, "Adaptive conformal inference under distribution shift," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 1660–1672.
- [45] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum, "Human-level concept learning through probabilistic program induction," *Science*, vol. 350, no. 6266, pp. 1332–1338, Dec. 2015.
- [46] S. A. Budenny et al., "Eco2AI: Carbon emissions tracking of machine learning models as the first step towards sustainable AI," in *Doklady Mathematics*, vol. 106. Cham, Switzerland: Springer, 2022, pp. S118–S128.
- [47] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, vol. 16, 2016, pp. 770–778.