

Knowledge Distillation for Sensing-Assisted Long-Term Beam Tracking in mmWave Communications

Mengyuan Ma, *Graduate Student Member, IEEE*, Nhan Thanh Nguyen, *Member, IEEE*, Nir Shlezinger, *Member, IEEE*, Yonina C. Eldar, *Fellow, IEEE*, A. Lee Swindlehurst, *Fellow, IEEE*, and Markku Juntti, *Fellow, IEEE*

Abstract—Infrastructure-mounted sensors can capture rich environmental information to enhance communications and facilitate beamforming in millimeter-wave systems. This work presents an efficient sensing-assisted long-term beam tracking framework that selects optimal beams from a codebook for current and multiple future time slots. We first design a large attention-enhanced neural network (NN) to fully exploit past visual observations for beam tracking. A convolutional NN extracts compact image features, while gated recurrent units with attention capture the temporal dependencies within sequences. The large NN then acts as the teacher to guide the training of a lightweight student NN via knowledge distillation. The student requires shorter input sequences yet preserves long-term beam prediction ability. Numerical results demonstrate that the teacher achieves Top-5 accuracies exceeding 93% for current and six future time slots, approaching state-of-the-art performance with a 90% complexity reduction. The student closely matches the teacher's performance while cutting complexity by another 90%, despite operating with 60% shorter input sequences. This improvement significantly enhances data efficiency, reduces latency, and lowers power consumption in sensing and processing.

Index Terms—Beam prediction, beam tracking, sensing, deep learning, knowledge distillation.

I. INTRODUCTION

Millimeter wave (mmWave) and terahertz (THz) communication combined with large-scale multiple-input multiple-output (MIMO) systems promise to achieve high data rates to meet the demand for emerging applications, such as vehicular networks, unmanned aerial vehicles, and augmented/virtual reality [1]. However, a large number of antennas is required to steer narrow focused beams toward the target users in order to mitigate the severe path loss and guarantee the desired quality of service. Accurate beam tracking and alignment are essential to maintain reliable links, especially in high-mobility scenarios where the rapid variation of the radio environment can cause tracking errors and wasted resources for frequent link reestablishment [2]. Nonetheless, conventional beam tracking methods are based on lengthy

codebook scanning, and thus typically incur significant overhead, posing challenges for real-time scenarios [3].

With the advancement of large-scale MIMO at high frequencies, integrated sensing and communications (ISAC) has emerged as a promising paradigm and brings new opportunities for more efficient beam tracking [4]. Environmental information from the target users' surroundings can be captured by various sensors and leveraged to facilitate communications by machine learning (ML) techniques. Such sensing-assisted data-driven methods can quickly adapt to environmental variations and were recently demonstrated to facilitate rapid and high-performance beamforming [5], [6]. This motivates the harnessing of sensory data to reduce beam training overhead and enable highly mobile mmWave communications systems.

A. Prior Work

Conventional beam tracking has been largely based on predefined codebooks [7]–[9]. The most accurate and straightforward beam training approach is an exhaustive search over all possible transmitter–receiver beam pair combinations. Alternatively, heuristic strategies such as hierarchical search [7], [9], two-stage search [8], and adaptive beamforming [10]–[15] have been explored. However, both exhaustive and heuristic search-based beamforming designs can incur significant training overhead and latency, particularly in large-scale MIMO systems. To address this challenge, deep learning-based beam training methods have been proposed [16]–[24]. In particular, deep learning plays a key role in exploiting sensing data captured by LiDAR [25], [26], radar [27], [28], cameras [3], [29]–[31], and global positioning systems (GPSs) [30] for beam management in communications. Such approaches fall under the category of sensing-assisted communications, which is among the main use cases in ISAC.

Beyond single-modality sensing, the joint exploitation of multiple sensing modalities provides enhanced and more robust performance [29], [32]. This approach is generally referred to as multimodal sensing-aided communications. Charan *et al.* [30] explored the use of both vision and GPS data for beam prediction, yielding better performance than when exploiting only one of the two modalities. LiDAR, radar, GPS, and cameras have been considered for joint beam prediction [33]–[38] using recurrent neural networks (RNNs) that leverage temporal dependencies [36] and for fusing data from distinct modalities using Transformers [33]–[35], [37]. Moreover, to enhance cross-modal feature extraction, Zhu *et al.* [39] designed a cross-attention module and used a dynamic fusion technique to improve beam prediction accuracy from radar and vision data. However, the models

This work was supported in part by the Research Council of Finland through 6G Flagship under Grant 346208 and through project DIRECTION under Grant 354901, EERA Project under Grant 332362, Infotech Oulu via EEWA Project, Business Finland, Keysight, MediaTek, Siemens, Ekahau, and Verkoton via project 6GLearn, and U.S. National Science Foundation grant CCF-2225575. The authors wish to acknowledge CSC – IT Center for Science, Finland, for computational resources. (Corresponding author: Mengyuan Ma.)

Mengyuan Ma, Nhan Thanh Nguyen, and Markku Juntti are with Centre for Wireless Communications, University of Oulu, P.O.Box 4500, FI-90014, Finland (e-mail: {mengyuan.ma, nhan.nguyen, markku.juntti}@oulu.fi). Nir Shlezinger is with School of ECE, Ben-Gurion University of the Negev, Beer-Sheva 84105, Israel (email: nirshl@bgu.ac.il). A. L. Swindlehurst is with the Dept. of Electrical Engineering & Computer Science, University of California, Irvine, CA, USA (e-mail: swindle@uci.edu). Yonina C. Eldar is with Faculty of Math and CS, Weizmann Institute of Science, Rehovot 76100, Israel (e-mail: yonina.eldar@weizmann.ac.il).

for effectively processing multimodal data inevitably result in high complexity algorithms, calling for efficient solutions.

A leading method in the ML literature to obtain efficient lightweight neural networks is based on knowledge distillation (KD) [40], which transfers the knowledge from a well-trained large “teacher” to a small “student” model while guaranteeing a desired level of performance, enabling a form of model compression [41]. KD is a regularization technique for transferring learned knowledge between classifiers, helping models to refine their predictions and enhance generalization [42]–[44]. This is particularly valuable when distilling knowledge from a large, accurate teacher model to a lightweight student for deployment on resource-limited devices [45]–[47]. By learning to match the teacher’s soft outputs, which convey nuanced information about class relationships and decision boundaries, the student can achieve competitive performance with significantly reduced computational cost. Moreover, even without a separate teacher, a model can leverage its own outputs or intermediate representations as internal guidance to improve learning, a process known as self-distillation or self-KD. In the context of beam tracking, [37] employed KD to obtain a compact model using only radar data, while in ISAC, KD has been applied to transceiver design [48], channel estimation and feedback [49]–[51], semantic communications [52], user positioning [53], and remote sensing [54], [55].

Despite the advances discussed above, most of the research pertinent to beam tracking focuses on predicting only the current beam based on current and/or past sensory data. Such methods result in frequent inference operations at each time step causing high overhead for sensing and processing in terms of power consumption, latency, and signaling. Long-term beam prediction can alleviate this by jointly predicting multiple future time steps, but this idea has remained largely unexplored. Existing work [25], [56] has considered long-term beam prediction based on vision and LiDAR data, respectively, achieving encouraging results. To facilitate learning from vision data, the convolutional neural network (CNN) model YOLOv4 [57] for object detection was used to extract the coordinates of potential targets from raw images in [56]. However, YOLOv4, which has approximately 6.4×10^7 parameters, is a relatively heavy model for deployment on resource-limited devices. More efficient long-term beam tracking requires further exploration.

B. Contributions

In this work, we develop a compact model that learns to implement long-term beam selection for current and multiple future time slots with the objective of maximizing the overall spectral efficiency. To this end, we cast the problem as an ML classification task, which we tackle in two stages. First, we initially ignore model complexity and sensing overhead, and design a sequence-to-sequence (Seq2Seq) neural network model that consists of dedicated CNNs and gated recurrent unit (GRU) networks with an additional attention mechanism. The designed CNNs together with a simple yet effective image pre-processing method can extract compact image features. Temporal dependencies across time slots are captured by the GRUs with attention, enabling robust long-term beam prediction.

In this paper, we use the complex pre-trained model to obtain a lightweight beam selection model. We identify KD as a practically suitable paradigm for this task since, as opposed to alternative model compression frameworks such as pruning and weight quantization [41], KD enables the compressed student model to notably deviate from how the teacher is structured. We thus employ the complex model to train a lightweight student that accepts shorter input sequences, reducing the model complexity and minimizing the operational cost associated with sensing and data processing.

The specific contributions of the paper are as follows:

- We develop an efficient end-to-end learning framework for long-term vision-based beam tracking by integrating CNNs, GRUs, and a multi-head attention (MHA) mechanism. This Seq2Seq model can effectively predict both current and future beams based on past sensor observations.
- We design a lightweight yet efficient student ML model for long-term beam tracking based on the KD technique. To this end, self-KD is leveraged to train a high-performing teacher model before imparting the knowledge to the student.
- We further enhance the student model to generate highly accurate beams with shorter input sequences. This significantly alleviates the computational burden, power consumption, and latency associated with continuous sensing data acquisition and processing.
- Finally, we perform extensive simulations based on a realistic dataset to demonstrate the proposed framework. The results show that the teacher model achieves over 93% Top-5 beam prediction accuracies across both current and six future time slots, approaching state-of-the-art performance with a 90% complexity reduction. Remarkably, the student model can perform similarly to the teacher despite employing 90% fewer parameters and 60% shorter input sequences.

The enablers of our contributions lies in the KD technique, advanced neural network structures, and efficient data preprocessing, which are judiciously designed for long-term beam tracking. Unlike [56], we design a dedicated CNN to extract representative semantic features from raw images and incorporate an attention mechanism to enhance the model’s ability to capture temporal dependencies within the input sequence. Furthermore, we develop a lightweight student model and train it using the KD technique, enabling the use of fewer past RGB frames for beam tracking. Although KD was used in [37] for developing a compact student model, the problem studied therein was the prediction of the current beam only, without considering the overhead for sensing and processing. To the best of the authors’ knowledge, this is the first work to optimize a learning framework for both computational and data efficiency in long-term sensing-aided beam tracking using KD. The specific differences between this paper and exiting works on sensing-aided beam tracking are summarized in Table I.

The rest of the paper is organized as follows. In Section II, we present the system model and problem formulation, and in Section III we introduce the long-term vision-based beam-

TABLE I. Comparison of this paper with prior work on sensing-aided beam tracking.

Reference(s)	Long-term Beam prediction	Reduce model complexity	Improve data efficiency
[26]–[31] [33]–[36] [3], [38], [39]	✗	✗	✗
[37]	✗	✓	✗
[56], [25]	✓	✗	✗
This work	✓	✓	✓

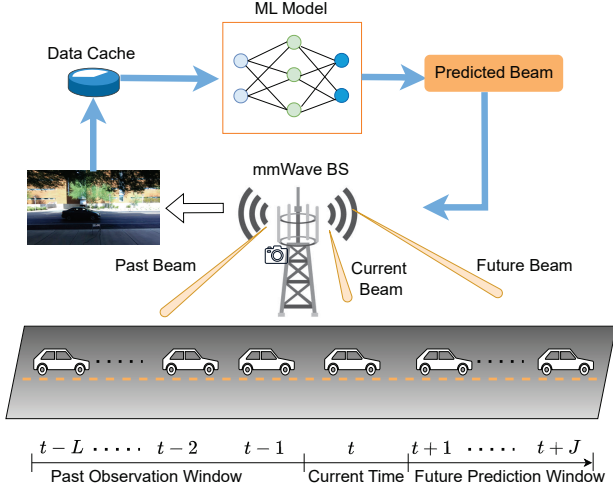


Fig. 1. Illustration of the considered system model. The BS senses the environment and the moving UE with an RGB camera. The sensory data are collected and cached for beam tracking using the designed ML model.

tracking design. We then delve into the KD-aided learning approach in Section IV. Finally, we provide simulation results and conclusions in Sections V and VI, respectively.

Throughout the paper, scalars, vectors, and matrices (or tensors) are denoted by lowercase, boldface lowercase, and boldface uppercase letters, respectively. The expectation operation is represented by $\mathbb{E}[\cdot]$. We use $|a|$ and $|\mathbf{A}|$ to denote the absolute value of a and the matrix (tensor) containing the absolute value of the entries of \mathbf{A} , respectively.

II. SYSTEM MODEL AND PROBLEM FORMULATION

A. System Model

We consider a downlink mmWave system, where the base station (BS) serves a single-antenna mobile user equipment (UE). The BS is equipped with a uniform linear array of N elements and an RGB camera (visual data sensor), as illustrated in Fig. 1. At time step t , the BS transmits $s[t] \in \mathbb{C}$ with normalized symbol power $\mathbb{E}[|s|^2] = 1$ to the UE. We assume a block fading channel between time slots. Let $\mathbf{v}[t] \in \mathbb{C}^N$ denote the beamforming vector at time step t with $\mathbb{E}[\mathbf{v}[t]^H \mathbf{v}[t]] = P$, where P is the transmit power budget. Then, the received signal $y[t]$ is given as

$$y[t] = \mathbf{h}[t]^H \mathbf{v}[t] s[t] + n[t], \quad (1)$$

where $\mathbf{h}[t] \in \mathbb{C}^N$ denotes the channel between the BS and the UE at time step t , and $n[t] \sim \mathcal{CN}(0, \sigma_n^2)$ is additive white

Gaussian noise (AWGN) with power σ_n^2 . The signal-to-noise ratio (SNR) at time slot t is thus given by

$$\text{SNR}[t] = \frac{|\mathbf{h}[t]^H \mathbf{v}[t]|^2}{\sigma_n^2}. \quad (2)$$

B. Problem Formulation

At the current time slot t , the goal is to have the BS determine the beamformers for the current and J future time slots, i.e., $\{t, t+1, \dots, t+J\}$. The spectral efficiency over these $J+1$ time slots is

$$R_J = \sum_{\tau=t}^{t+J} \log(1 + \text{SNR}[\tau]). \quad (3)$$

Let $\mathcal{V} = \{\mathbf{v}_1, \dots, \mathbf{v}_{|\mathcal{V}|}\}$ and $\mathcal{I}_{\mathcal{V}} = \{1, \dots, |\mathcal{V}|\}$ denote the beamforming codebook and its associated index set. The beam tracking problem is expressed as

$$\underset{\mathbf{v}[\tau] \in \mathcal{V}, \forall \tau}{\text{maximize}} \quad R_J. \quad (4)$$

For low SNR scenarios, we can approximate R_J and reformulate problem (4) as [3], [25], [58]

$$\underset{\mathbf{v}[\tau] \in \mathcal{V}, \forall \tau}{\text{maximize}} \quad \sum_{\tau=t}^{t+J} |\mathbf{h}[\tau]^H \mathbf{v}[\tau]|^2. \quad (5)$$

Let $\mathbf{b}^*[t] = [b^*[t], b^*[t+1], \dots, b^*[t+J]]^T$ represent the vector of beam indices corresponding to the optimal solution of (5), i.e., $b^*[t] = \arg \max_{b[t] \in \mathcal{I}_{\mathcal{V}}} |\mathbf{h}[t]^H \mathbf{v}_{b[t]}|^2$. Then problem (5) can be expressed as

$$\mathbf{b}^*[t] = \arg \max_{b[\tau] \in \mathcal{I}_{\mathcal{V}}, \forall \tau} \sum_{\tau=t}^{t+J} |\mathbf{h}[\tau]^H \mathbf{v}_{b[\tau]}|^2. \quad (6)$$

The optimal solution to (6) can be obtained by decoupling it into $J+1$ subproblems with each solved via an exhaustive search over the $|\mathcal{V}|$ candidate beams. However, the complexity of such a method scales as $J|\mathcal{V}|$, which can incur high latency, especially with the large codebooks used in massive MIMO systems. Moreover, this approach requires perfect channel state information (CSI) at not only the current time slot, but also the J future time slots, which is generally unavailable in practice.

In this work, we consider CSI-free beam tracking, where instead of aiming to recover $\mathbf{b}[t]$ based on knowledge of $\mathbf{h}[t]$, we utilize sensed visual data, denoted by $\mathbf{Z}[t]$. Accordingly, our aim is to design a CSI-free mapping from $\mathbf{Z}[t]$ into $\mathbf{b}[t]$, such that the results remains effective with respect to the CSI-based performance measure in (6). Unlike [3], [27], [29]–[31], [33]–[39] which address problem (6) by decoupling it into $J+1$ subproblems, we propose an efficient learning framework that directly solves problem (6) for long-term beam tracking, as will be elaborated below.

III. VISION-BASED LONG-TERM BEAM TRACKING

A. ML Task Definition

Let $\mathbf{Z}[t] \in \mathbb{R}^{3 \times d_H \times d_W}$ denote the RGB image obtained at time slot t , where the dimension 3 corresponds to the number of RGB channels, and d_H and d_W respectively represent the image height and width in pixels. Let $\mathcal{Z}[t]$ denote the sequence of sensory data, i.e., RGB images, from

the L previous time slots to the current time t , defined by $\mathcal{Z}[t] = \{\mathbf{Z}[t-L], \mathbf{Z}[t-L+1], \dots, \mathbf{Z}[t]\}$. The objective of the learning task is to predict the optimal beams (equivalently the optimal beam indices in \mathcal{I}_V) for the current time slot t and the J future time slots $t+1, \dots, t+J$. This problem can be cast as an ML classification task, where the number of classes C is the size of the codebook, i.e., $C = |\mathcal{V}|$.

Denote the data preprocessing operations by $\mathcal{X}[t] = g(\mathcal{Z}[t])$, mapping the input sequence to the ML model. Let $f(\mathcal{X}[t]; \Theta)$ denote the ML model with learnable parameters Θ . The ML model outputs the probabilities of all possible beams at the $J+1$ (current and future) time slots. Let $p_c[t+j]$ denote the probability of selecting the c -th beam in the codebook at time slot $t+j$, and define $\mathbf{p}[t+j] = [p_1[t+j], \dots, p_C[t+j]]^\top \in \mathbb{R}^C, j = 0, \dots, J$. The intended output of the ML model is a probability distribution matrix given by

$$f(\mathcal{X}[t]; \Theta) = [\mathbf{p}[t], \dots, \mathbf{p}[t+J]] \triangleq \mathbf{P}[t] \in \mathbb{R}^{C \times (J+1)}. \quad (7)$$

The predicted beam index is obtained as

$$\hat{b}[\tau] = \arg \max_{c \in \mathcal{I}_V} p_c[\tau], \quad \tau = t, \dots, t+J. \quad (8)$$

The desired ML model for vision-aided beam tracking can be written as

$$f^*(\cdot; \Theta^*) = \arg \max_{f(\cdot; \Theta)} \sum_{\tau=t}^{t+J} \mathbb{P}\{\hat{b}[\tau] = b^*[\tau]\}, \quad (9)$$

where $\mathbb{P}\{\cdot\}$ denotes the probability. We note that J and L are hyperparameters, which are determined empirically.

B. Data Preprocessing

The raw images captured by the camera contain rich information about the environment surrounding the mobile UE. However, directly using these raw images as input to the ML model poses significant challenges for efficient learning, as irrelevant information for beam tracking acts as interference and increases the computational burden, motivating the need for effective preprocessing techniques. One approach is to extract bounding boxes of potential objects using additional CNN-based detectors, as demonstrated in [3], [56]. However, this approach introduces significant computational complexity. For instance, the well-known YOLOv4 model contains approximately 6.4×10^7 parameters.

To overcome the complexity challenge, we adopt background subtraction [33] to remove background and irrelevant information without incurring additional computational overhead. Specifically, the data preprocessing operations $g(\cdot)$ consist of three steps:

- **Step 1:** First, we resize and transform the raw RGB images to grayscale. This step effectively diminishes the dimensions of the trainable tensors, reducing the complexity of the ML model. We denote

$$\mathcal{Z}'[t] = \{\mathbf{Z}'[t-L], \mathbf{Z}'[t-L+1], \dots, \mathbf{Z}'[t]\} \quad (10)$$

as the sequence of post-processed images with $\mathbf{Z}'[\tau] \in \mathbb{R}^{d'_H \times d'_W}, \forall \tau$ ($d'_H < d_H, d'_W < d_W$).

- **Step 2:** Based on $\mathcal{Z}'[t]$ in (10), we further construct

$$\mathcal{X}'[t] = \{\mathbf{X}'[t-L+1], \mathbf{X}'[t-L+2], \dots, \mathbf{X}'[t]\}, \quad (11)$$

where $\mathbf{X}'[t-l] = |\mathbf{Z}'[t-l] - \mathbf{Z}'[t-l-1]|, l = 0, \dots, L-1$ represents the *difference image* that highlights moving objects. This operation can effectively remove interference due to static objects, i.e., the background noise. However, useful information pertinent to the moving UE can also be compromised.

- **Step 3:** To enhance the difference image, we construct a sequence of motion masks, defined as

$$\mathcal{X}[t] = \{\mathbf{X}[t-L+1], \mathbf{X}[t-L+2], \dots, \mathbf{X}[t]\}, \quad (12)$$

which are obtained by setting the large values in $\mathbf{X}'[\tau]$ above a given threshold equal to one, and setting all others to zero. Neglecting the time stamp τ , the motion mask is given by:

$$\mathbf{X}(m, n) = \begin{cases} 1 & \text{if } \mathbf{X}'(m, n) \geq \varepsilon \max(\mathbf{X}'), \\ 0 & \text{otherwise,} \end{cases} \quad (13)$$

where $\mathbf{X}(m, n)$ denotes the entry at the m -th row and n -th column of \mathbf{X} , and $\max(\mathbf{X}')$ returns the maximum value in \mathbf{X}' . Here, $\varepsilon \in (0, 1)$ controls the percentage of information retained about the UE. A small ε preserves most of this information and thus we set it to 0.1 in the subsequent numerical experiments in Section V.

We will illustrate the efficiency of this preprocessing procedure via simulation results in Section V.

C. ML Model Design

1) *Motivation:* As seen in (9), the ML task is to predict a sequence of beam selections over current and future time steps based on a time series sequence of past images. Such Seq2Seq learning tasks have been widely studied in natural language processing and computer vision. Well-known techniques such as RNNs [59] and attention-based mechanisms [60]–[62] have been developed for text and video generation. Transformer architectures [60] have achieved remarkable performance for large language models (LLMs), such as GPT [61], and have been leveraged for multimodal sensing tasks [33], [37], [39]. Although Transformer-based ML models are powerful for capturing long-term dependencies between tokens, they have high complexity.

Alternatively, we seek to design an efficient ML model employing RNN architectures and an attention mechanism. In particular, we consider using a GRU network [63], which is a variant of the conventional RNN architecture. GRUs address the limitations of standard RNNs by incorporating gating mechanisms that regulate the flow of information, making them more effective at capturing long-term dependencies in sequential data.

2) *Model Structures:* Fig. 2 illustrates the considered GRU-based Seq2Seq model structure, which consists of three functional blocks: embedding, GRU, and prediction. The embedding block extracts features of the high-dimensional inputs $\mathbf{X}[t-l], l = 0, \dots, L-1$ to form low-dimensional feature vectors $\mathbf{f}[t-l] \in \mathbb{R}^D, l = 0, \dots, L-1$ in a latent space of size D . These feature vectors are then processed by the GRUs to obtain their sequential information, which is summarized by the output representation feature vectors $\mathbf{f}'[t+j] \in \mathbb{R}^{D'}, j = 0, \dots, J$. The prediction block further

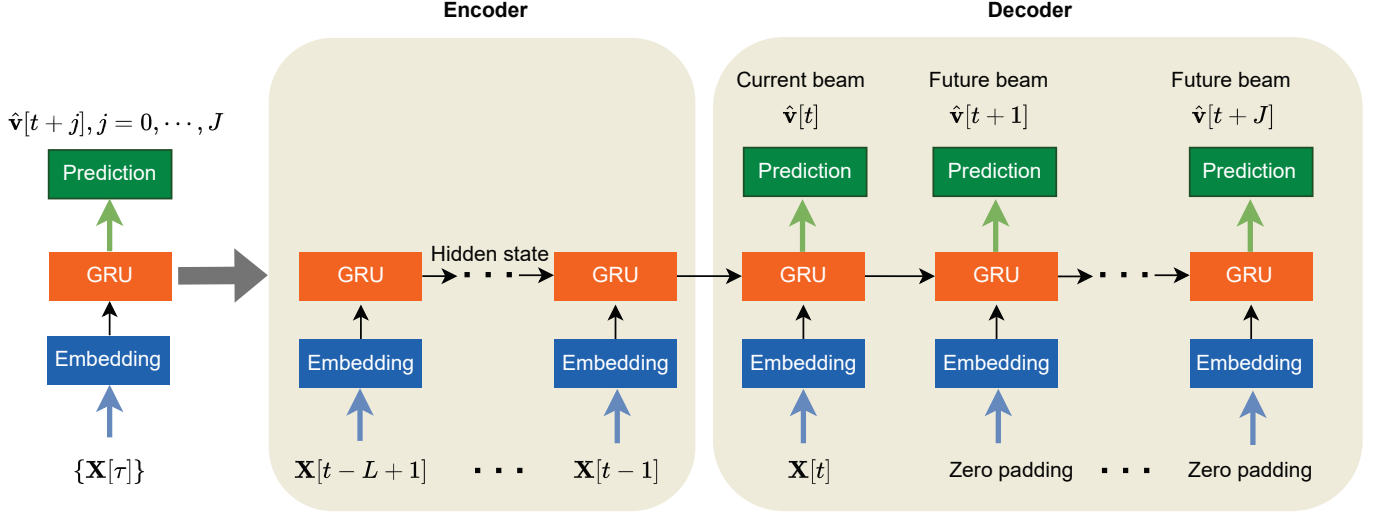


Fig. 2. Illustration of the ML model.

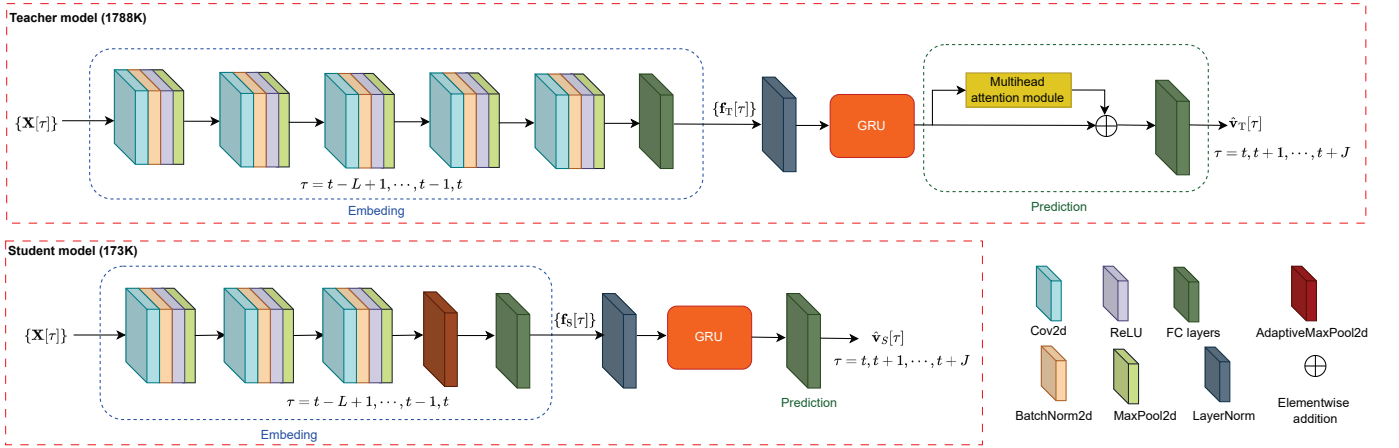


Fig. 3. Illustration of the teacher and student model structures.

processes the representation features and outputs the predicted beam vector $\hat{v}[t+j], j = 0, \dots, J$. The entire process can be divided into two parts, i.e., the encoder and decoder. The encoder processes the input sequence $\{X[\tau]\}$ (i.e., $\mathcal{X}[t]$) step by step, updating its hidden state and ultimately producing a context vector that summarizes the information of the entire input. This context vector is then passed to the decoder, which generates the representation feature vectors $f'[t+j] \in \mathbb{R}^{D'}$, $j = 0, \dots, J$ one by one, given the current data $X[t]$. At each step, the decoder produces the current representation feature based on the previous hidden state. The hidden state in the GRUs helps the model efficiently capture temporal dependencies among the input features.

To find a performance guarantee, we first design a large model that has powerful embedding and prediction blocks. A simple lightweight model will be trained with the guidance of the large model leveraging the KD technique. To distinguish them, the large and lightweight models are referred to as the teacher and student models, respectively. These models are illustrated in Fig. 3. A pretrained ResNet for image feature

extraction [33]–[37] would result in high computational complexity and memory usage. To improve efficiency, we design dedicated CNN networks to extract spatial features of the images in the input sequence. Specifically, the embedding block of the teacher model consists of a five-layer CNN and a flattening layer. In contrast, the student model contains only three conventional layers with an *AdaptiveMaxPool* module to reduce the variable dimension. The prediction block of the student model is a multilayer perceptron (MLP), whereas that of the teacher model includes an additional residual multihead attention (MHA) module.

The MHA aims to provide a self-attention mechanism that allows each feature in a sequence to attend to all others and simultaneously learn different contextual information components. For the mathematical foundations of MHA, we refer the reader to [60]. Using MHA self-attention after the GRU combines the strengths of both sequential and attention-based modeling. Specifically, while GRUs effectively capture temporal dependencies in a step-by-step manner, they focus mainly on local dependencies. In contrast, MHA self-attention

allows the model to directly attend to all positions in the sequence, enhancing its ability to capture global features. Furthermore, multiple attention heads enable the extraction of diverse features from the GRU outputs, leading to richer and more informative representations that can improve the model's expressive ability and boost performance. The effectiveness of the MHA module will be justified using numerical experiments in Section V.

IV. KNOWLEDGE DISTILLATION AIDED LEARNING

As discussed in Section III, the teacher model employs a more complex architecture than the student model and, thus can achieve superior performance in challenging beam prediction tasks. However, this performance gain comes at the cost of high computational complexity and a dependence on long input sequences spanning multiple time slots. These requirements lead to increased power consumption, higher latency, and greater overhead for data collection and preprocessing, posing significant challenges for practical deployments, especially in resource-constrained environments.

While conventional model compression techniques such as pruning [64] or quantization [65] can reduce model complexity, they fail with shorter input sequences and thus are not useful for latency reduction. In contrast, KD provides both a compression mechanism and a learning framework, enabling the student model to inherit the predictive capability of the high-performing teacher model while being explicitly trained to function with shorter sequences. This makes KD particularly well-suited for our objective of achieving both low complexity and low-latency beam tracking. We elaborate on the use of KD for efficient long-term beam tracking below.

A. KD Loss Function

The loss function plays a crucial role in KD-based training of the student model. In this section, we first present the details of the loss function employed to train the student model for beam tracking, given that the teacher model has already been developed as described in the previous section.

Let $\mathcal{L}_{\text{task}}$ and $\mathcal{L}_{\text{distill}}$ denote the task loss computed from the dataset and the distillation loss arising from the disparity between the teacher and student, respectively. In typical KD, the student minimizes a weighted combination of these two components:

$$\mathcal{L}_s = (1 - \beta)\mathcal{L}_{\text{task}} + \beta\mathcal{L}_{\text{distill}}. \quad (14)$$

Here, a trade-off parameter $\beta \in [0, 1]$ is employed to control the balance between these components: $\beta = 0$ corresponds to learning purely from hard labels, whereas $\beta = 1$ corresponds to learning solely from the teacher. While a large β may lead the student to better mimic the teacher, it does not necessarily guarantee improved generalization. In fact, excessive reliance on the teacher can cause overfitting and degrade learning performance [66]. The value of β is typically determined empirically. In the following, we formulate the specific loss components $\mathcal{L}_{\text{task}}$ and $\mathcal{L}_{\text{distill}}$ for the beam tracking task.

1) *Task Loss*: Let $\mathcal{D} = \{\{\mathcal{Z}[t], \mathbf{b}^*[t]\}, t = 0, \dots, T\}$ denote the set of data sequences from the source dataset, where T denotes the number of time slots over which vision data are collected, and $\mathcal{Z}[t]$ and $\mathbf{b}^*[t]$ are the input and label of the ML model, respectively. The received signal $y[t]$ in (1) is leveraged to obtain $\mathbf{b}^*[t]$, which in mmWave communications is generally distributed non-uniformly among the C candidate beams. Such a class imbalance among the datasets can lead to poor performance for the minority class. During training, we use the Focal loss [67] for $\mathcal{L}_{\text{task}}$, which is a modification of the standard cross-entropy loss designed to address the class imbalance problem.

For the τ -th sample $\mathbf{Z}[\tau]$, the output of the ML model is the predicted probability vector $\mathbf{p}[\tau]$ for the C beam classes. Define the softmax function $\sigma(\mathbf{x})$ for a vector $\mathbf{x} = [x_1, \dots, x_N]$ as

$$\sigma_i(\mathbf{x}) = \frac{\exp(x_i)}{\sum_{n=1}^N \exp(x_n)}, \quad (15)$$

and let $\mathbf{z}[\tau] = [z_1[\tau], \dots, z_C[\tau]]$ denote the vector of output logits. Then the c -th element of $\mathbf{p}[\tau]$ is obtained as $p_c[\tau] = \sigma_c(\mathbf{z}[\tau])$. The Focal loss for a single sample $\mathbf{Z}[\tau]$ is given by

$$l_{\text{Focal}}[\tau] = -\alpha(1 - p_{b^*}[\tau])^\gamma \log(p_{b^*}[\tau]), \quad (16)$$

where $p_{b^*}[\tau]$ denotes the predicted probability of selecting the ground-truth beam index $b^*[\tau]$ at time slot τ . The hyperparameter α is the weighting factor addressing class imbalance, and γ is the focusing parameter that demphasizes easy examples. A large γ leads to a small loss for well-classified samples, i.e., those with high output probabilities. This helps the model focus on difficult or misclassified samples, which are more informative. On the contrary, $\gamma = 0$ leads to the conventional cross-entropy loss which treats all samples with equal importance. The overall task loss for the input sequence $\mathcal{Z}[t]$ is expressed as

$$\mathcal{L}_{\text{task}}[t] = \sum_{\tau=t}^{t+J} l_{\text{Focal}}[\tau]. \quad (17)$$

2) *Distillation Loss*: We adopt the Kullback-Leibler (KL) divergence as the distillation loss $\mathcal{L}_{\text{distill}}$, which measures the similarity between the output distributions of the teacher and student models, respectively denoted as $\tilde{P}_{\text{teacher}}^{(\tau)}$ and $\tilde{P}_{\text{student}}^{(\tau)}$, given the input sample $\mathbf{Z}[\tau]$. The KL divergence is computed as

$$D_{\text{KL}}\left(\tilde{P}_{\text{teacher}}^{(\tau)} \parallel \tilde{P}_{\text{student}}^{(\tau)}\right) = \sum_{c=1}^C \tilde{P}_{\text{teacher}}^{(\tau,c)} \log\left(\frac{\tilde{P}_{\text{teacher}}^{(\tau,c)}}{\tilde{P}_{\text{student}}^{(\tau,c)}}\right), \quad (18)$$

where $\tilde{P}_{\text{teacher}}^{(\tau,c)} = \sigma_c(\mathbf{z}_{\text{teacher}}[\tau]/\Gamma)$, $\mathbf{z}_{\text{teacher}}[\tau]$ is the vector of output logits from the teacher model, and $\tilde{P}_{\text{student}}^{(\tau,c)}$ is similarly defined. Here, Γ represents the temperature used to control the smoothness of the distribution; increasing Γ makes the distribution more uniform, while $\Gamma \rightarrow 0$ results in the one-hot distribution. Therefore, an appropriate value for Γ is needed to enable the student model to learn well from the teacher, and can be determined empirically. The distillation

loss for the input sequence $\mathcal{Z}[t]$ is given by

$$\mathcal{L}_{\text{distill}}[t] = \sum_{\tau=t}^{t+J} D_{\text{KL}} \left(\tilde{P}_{\text{teacher}}^{(\tau)} \| \tilde{P}_{\text{student}}^{(\tau)} \right) \cdot \Gamma^2, \quad (19)$$

where the multiplication by Γ^2 arises since the gradients produced by the softmax function are scaled by $1/\Gamma$.

B. Training Procedure

Algorithm 1 summarizes the KD-aided learning procedure, where $f_{\text{T}}(\cdot; \Theta_{\text{T}})$ represents the pretrained teacher model; \mathcal{D}_{tr} and \mathcal{D}_{evl} denote the training and validation datasets obtained from the overall dataset \mathcal{D} without any overlap; E represents the number of total epochs and N_{b} denotes the number of batches in each epoch. To begin the training, the model parameters are randomly initialized. For each epoch, N_{b} batches are generated by randomly dividing \mathcal{D}_{tr} into batches of predefined size B . Steps 4–15 update the model parameters in a batch training manner. Data preprocessing is first performed in step 5, where $\mathcal{T}^{(n)} = \{t_i^{(n)} | \mathcal{Z}[t_i^{(n)}] \in \mathcal{D}_{\text{tr}}^{(n)}, \forall i\} = \{t_1^{(n)}, \dots, t_B^{(n)}\}$ denotes the set of time stamps in the n -th batch. The q -th sequence sample $\{\mathcal{X}[t_q^{(n)}]\}$ in the n -th batch is fed into the embedding block, which returns the low-dimensional feature vectors $\{\mathbf{f}[t_q^{(n)}]\}$. After the GRU and prediction modules, the ML model outputs the predicted probability matrix $\mathbf{P}[t_q^{(n)}]$, which is leveraged to compute the sample task loss and distillation loss as indicated by steps 9 and 10, respectively.

Next, the average task and distillation loss over a batch for the considered $J+1$ time slots are computed in steps 12 and 13. The overall loss is obtained and used to optimize the model parameters by the backpropagation algorithm in step 14. When batch learning is completed, the performance of the model $f_{\text{S}}(\cdot; \Theta_{\text{S}})$ is evaluated on the validation dataset \mathcal{D}_{evl} in step 16. The optimal model parameters Θ_{S}^* are updated if a lower validation loss is found, as shown in steps 17–20. The optimal model parameters Θ_{S}^* are returned when the maximum number of epochs is reached or the best validation loss L_{evl}^* stops decreasing over a predefined number of consecutive epochs.

To enhance the learning efficiency of the student model via KD, a good teacher is required. We first train the teacher model by setting $\beta = 0$ in Algorithm 1, and then refine it with self-KD. With the refined teacher model, we train the student model with KD. The effectiveness of the proposed KD-aided learning framework will be experimentally validated in the sequel.

V. NUMERICAL RESULTS

In this section, we provide extensive numerical simulations to demonstrate the performance of the proposed sensing-assisted long-term beam tracking approach¹. Experiments are based on Scenario 9 of the DeepSense 6G dataset [68], which provides sensory data and optimal beams for real-world mmWave communications.

Algorithm 1: KD-Aided Learning for Problem (9).

Input: Training and validation datasets $\mathcal{D}_{\text{tr}}, \mathcal{D}_{\text{evl}}$; Pretrained teacher model $f_{\text{T}}(\cdot; \Theta_{\text{T}})$

Output: Student model parameters Θ_{S}

- 1 Initialize $\Theta_{\text{S}}^*, \Theta_{\text{S}} = \Theta_{\text{S}}^*, \mathcal{L}_{\text{evl}}^* = 1000, \beta, \Gamma$, and learning rate.
- 2 **for** $e = 1, \dots, E$ **do**
- 3 Randomly divide \mathcal{D}_{tr} into N_{b} batches $\{\mathcal{D}_{\text{tr}}^{(n)}\}_{n=1}^{N_{\text{b}}}$ with batch size B .
- 4 **for** $n = 1, \dots, N_{\text{b}}$ **do**
- 5 Perform data preprocessing
- 6 $\mathcal{X}[t] = g(\mathcal{Z}[t])$ with $t \in \mathcal{T}^{(n)} = \{t_1^{(n)}, \dots, t_B^{(n)}\}$
- 7 **for** $q = 1, \dots, B$ **do**
- 8 Obtain embedded feature $\{\mathbf{f}[t_q^{(n)}]\}$ for $\{\mathcal{X}[t_q^{(n)}]\}$.
- 9 Feed $\{\mathbf{f}[t_q^{(n)}]\}$ into the GRU module with postprocessing of the prediction module. Compute the student model output $\mathbf{P}[t_q^{(n)}] = f_{\text{S}}(\{\mathcal{X}[t_q^{(n)}]\}; \Theta_{\text{S}})$.
- 10 Compute the task loss $\mathcal{L}_{\text{task}}[t_q^{(n)}]$ in (17).
- 11 Compute the distillation loss $\mathcal{L}_{\text{distill}}[t_q^{(n)}]$ in (19) based on $f_{\text{S}}(\{\mathcal{X}[t_q^{(n)}]\}; \Theta_{\text{S}})$ and $f_{\text{T}}(\{\mathcal{X}[t_q^{(n)}]\}; \Theta_{\text{T}})$.
- 12 **end**
- 13 Compute average task loss over the batch in $J+1$ time slots: $\mathcal{L}_{\text{task}} = \frac{1}{B(J+1)} \sum_{t \in \mathcal{T}^{(n)}} \mathcal{L}_{\text{task}}[t]$.
- 14 Compute average distillation loss over the batch in $J+1$ time slots: $\mathcal{L}_{\text{distill}} = \frac{1}{B(J+1)} \sum_{t \in \mathcal{T}^{(n)}} \mathcal{L}_{\text{distill}}[t]$.
- 15 Obtain the overall loss in (14) and update Θ_{S} with an optimizer.
- 16 **end**
- 17 Compute validation loss $\mathcal{L}_{\text{evl}}^{(e)}$ based on $f_{\text{S}}(\cdot; \Theta_{\text{S}})$ and \mathcal{D}_{evl} .
- 18 **if** $\mathcal{L}_{\text{evl}}^{(e)} < \mathcal{L}_{\text{evl}}^*$ **then**
- 19 update the best model $\Theta_{\text{S}}^* = \Theta_{\text{S}}$.
- 20 update the best loss $\mathcal{L}_{\text{evl}}^* = \mathcal{L}_{\text{evl}}^{(e)}$.
- 21 **end**
- 22 **Return** Θ_{S}^* .

A. Dataset Preparation

Fig. 4 illustrates Scenario 9 from the DeepSense 6G dataset. A BS equipped with a 16-element ULA and an RGB camera is deployed roadside to receive signals from a moving UE, which transmits signals at 60 GHz using a quasi-omni antenna. An oversampled beamforming codebook with 64 predefined beams is used at the BS. At each time step of duration of 128 ms, the BS performs beam sweeping to measure the received power across all beams and captures an RGB image of the UE. The channel coherence time is approximately 143 ms [25], which is longer than the sampling interval, justifying the assumption of a block fading channel in Section II-A.

Scenario 9 of the DeepSense 6G dataset contains 5964 samples that belong to multiple data sequences. In a data sequence, the moving UE passes by the BS, which acquires RGB images and the received power from each beam at multiple time steps. For any time step t , the maximum number of past images is set to $L = 8$ in each sequence sample $\mathcal{Z}[t]$, while the number of future time steps for beam prediction is set to $J = 6$. Therefore, the useful dataset contains a total of

¹The simulation codes are available at <https://github.com/WillysMa/Sensing-Assisted-Beam-Tracking.git>.

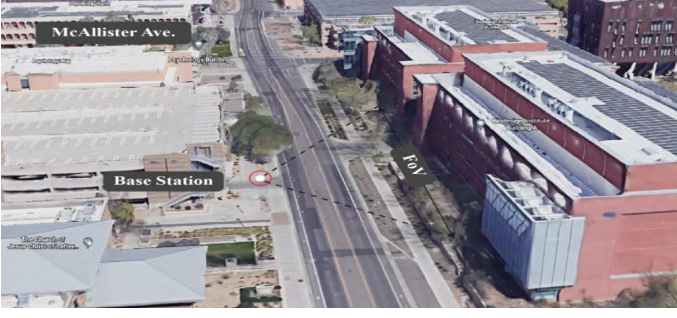


Fig. 4. Scenario 9 of the DeepSense 6G dataset.

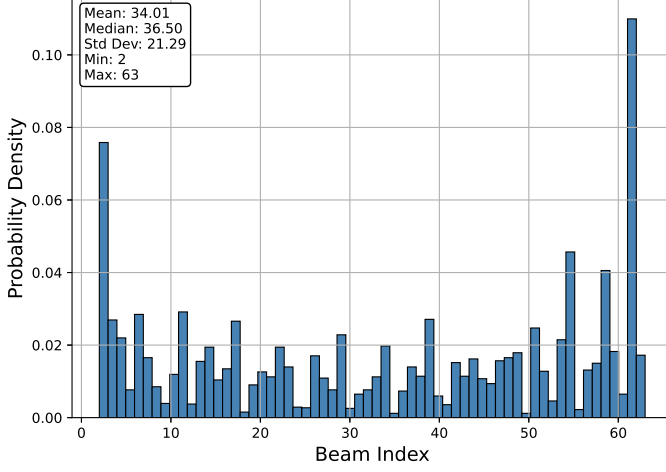


Fig. 5. Statistics of the optimal beam index in the considered dataset.

$T = 4060$ samples $\{\mathcal{Z}[t], \mathbf{b}^*[t]\}$ to guarantee that the images and labels are within the same data sequence. The training and validation datasets consist of 80% and 20% of the total 4060 samples, which corresponds to 3286 and 774 samples, respectively. Due to the limited data, the validation dataset is also used at the inference stage to evaluate the generalization performance of the ML models. Fig. 5 shows the statistics of the dataset, demonstrating that the numbers of samples belonging to different classes are imbalanced. Therefore, it is reasonable to use the Focal loss in (16) for training models.

B. Experiment Setup

The ML models are implemented using PyTorch and trained on NVidia Tesla V100 GPUs. In the training stage, we set the batch size to $B = 32$ and the maximum number of epochs to $E = 100$. The initial learning rate is 10^{-4} and a cyclic cosine annealing scheduler is used. We set $\alpha = 1$ and $\gamma = 2$ for the Focal loss function. To prevent overfitting, we use a weight decay of 10^{-4} and clip the gradient to be no more than 10. Furthermore, an early stopping technique is adopted to improve training efficiency; in particular, the training process is terminated once the validation loss stops decreasing over 20 consecutive epochs or the maximum number of epochs is reached.

The hidden size of the GRUs and the feature size are set to 64 for both the teacher and student models. However, the teacher model uses a two-layer GRU network, while only a single-layer GRU network is employed in the student model.

In the teacher model, we use $I = 8$ heads for diverse attentions and five CNN layers for feature extraction. Only three CNN layers are used for the student model. As a result, the teacher model has approximately 1.8×10^6 total trainable parameters, compared to only 1.7×10^5 trainable parameters for the student model, representing a complexity reduction of over 90%.

Fig. 6 illustrates the data preprocessing operation $g(\cdot)$ with $L = 8$. Fig. 6(a)–(c) shows a sequence of 8 raw images, and the 7 corresponding difference and motion masks, respectively. It can be seen that the motion masks highlight the moving UE while effectively removing the background noise.

C. Performance Metrics

The task loss at the inference stage reflects the overall generalization performance. Besides the task loss, we adopt the Top- K accuracy to measure whether the ground-truth label is among the model's Top- K predicted labels. For the $M = 774$ validation samples, let $\hat{y}_{m,1}, \hat{y}_{m,2}, \dots, \hat{y}_{m,K}$ be the Top- K predicted classes (e.g., highest K logits) for sample m . The Top- K accuracy is then defined as

$$\text{Top-}K \text{ Accuracy} = \frac{1}{M} \sum_{m=1}^M \mathbf{1}(y_m \in \{\hat{y}_{m,1}, \dots, \hat{y}_{m,K}\}), \quad (20)$$

where y_m denotes the ground-truth label, and $\mathbf{1}(\cdot)$ is the indicator function (1 if true, 0 otherwise). The Top- K score is based on “hard” decisions, which may be unnecessary in practice. Given that the close-to-optimal beams may be sufficient for guaranteeing the desired quality of service, a distance-based accuracy (DBA) metric is introduced for the beam prediction task [69], which assigns a score based on the distance between the predicted and ground-truth beams. Specifically, based on the Top-3 predicted beams, the DBA score is given by

$$\text{DBA} = \frac{1}{3} \sum_{k=1}^3 Y_k \quad (21)$$

where

$$Y_k = 1 - \frac{1}{M} \sum_{m=1}^M \min_{1 \leq i \leq k} \min \left(\frac{|\hat{y}_{m,i} - y_m|}{\Delta}, 1 \right), \quad (22)$$

and Δ is a normalization factor determining the maximum tolerable distance between the optimal and predicted beams. The term $\min \left(\frac{|\hat{y}_{m,i} - y_m|}{\Delta}, 1 \right)$ serves as a normalized penalty. A small Δ will induce a large penalty near 1, making the DBA more sensitive to prediction errors. On the contrary, a large Δ indicates tolerance for larger deviations before the penalty is capped at 1. Unless otherwise mentioned, we set $\Delta = 5$ for performance evaluation [69].

Note that both the Top- K and DBA scores target only one time slot. To reflect the overall performance across all $J + 1$ time slots, we further define the average Top- K (ATop- K) and average DBA (ADBA) over all time slots, given by

$$\text{ATop-}K \text{ Accuracy} = \frac{1}{J+1} \sum_{\tau=t}^{t+J} \text{Top-}K[\tau], \quad (23)$$

$$\text{ADBA} = \frac{1}{J+1} \sum_{\tau=t}^{t+J} \text{DBA}[\tau], \quad (24)$$

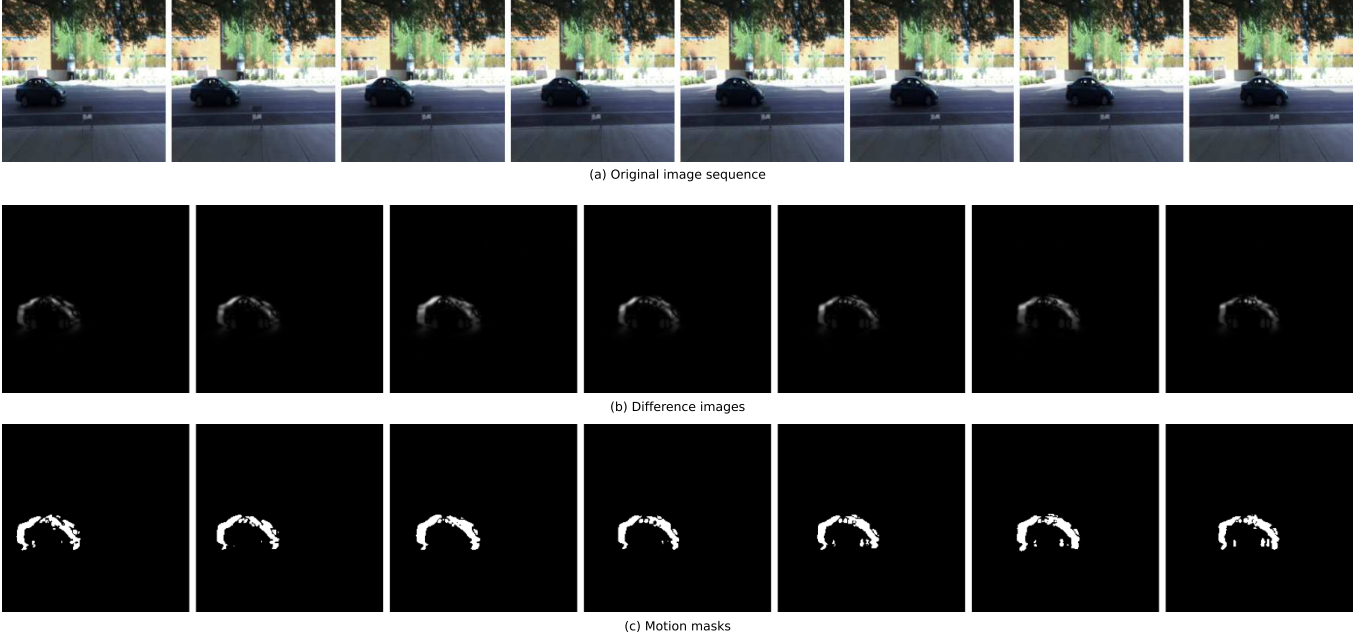


Fig. 6. Illustration of data preprocessing with $L = 8$. (a): The original sequence of 8 images. (b): The sequence of 7 difference images (c): The sequence of 7 motion masks.

TABLE II. Overall generalization performance of the teacher model in % for ATop- k accuracy and ADBA score.

Metric	W/o MHA	With MHA	Self-KD	Optimal [56]
Test loss	1.141	1.050	1.016	0.8158
ATop-1	40.77	42.91	44.94	50.20
ATop-3	77.44	79.97	81.45	87.15
ATop-5	92.31	93.35	94.63	96.81
ADBA	93.50	94.47	95.00	96.63

TABLE III. Overall generalization performance of the student model in % for ATop- k accuracy and ADBA score.

Metric	Methods	$L = 8$	$L = 5$	$L = 3$
Test loss	W/o KD	1.342	1.458	1.582
	With KD	1.045	1.178	1.283
ATop-1	W/o KD	38.34	33.89	33.19
	With KD	44.36	42.88	39.52
ATop-3	W/o KD	71.98	67.15	64.86
	With KD	80.23	78.13	73.13
ATop-5	W/o KD	88.32	85.68	81.45
	With KD	93.62	91.90	90.57
ADBA	W/o KD	90.31	87.66	85.12
	With KD	94.86	93.28	91.56

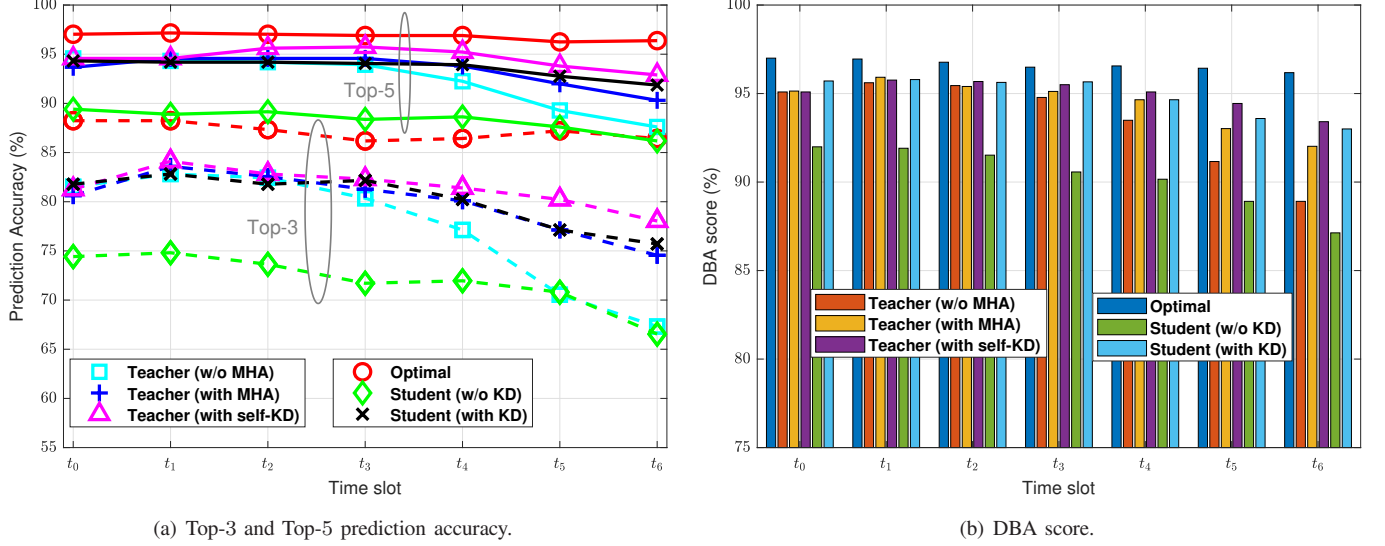
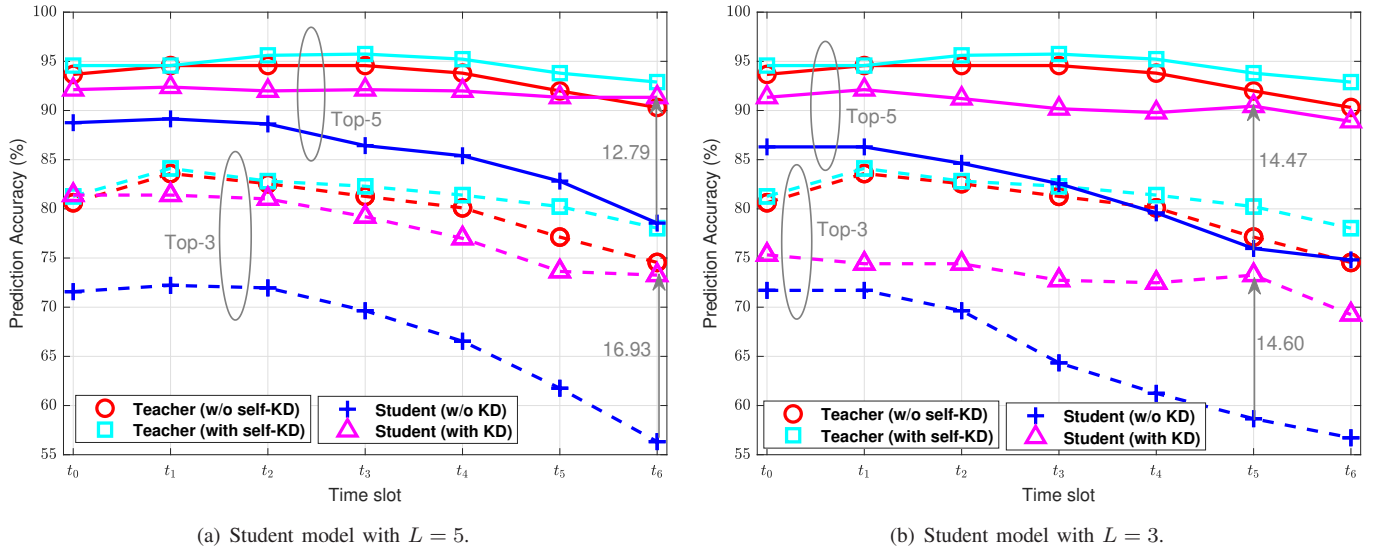
where $\text{Top-}K[\tau]$ and $\text{DBA}[\tau]$ represent the Top- K accuracy and DBA score at time slot τ , respectively.

D. Overall Performance

Table II summarizes the overall generalization performance of the teacher model, where “W/o MHA” and “With MHA” represent the architecture without and with the MHA module, respectively. In the self-KD scheme, the teacher model

contains the MHA module and is trained with guidance from the architecture “With MHA”. In [56], Jiang *et al.* first obtain the coordinates of potential sensing targets from images with YOLOv4 [57] and manually selects the correct target to be sensed as the input of the neural network consisting of GRUs and MLPs. Since the interference has been removed from the dataset before training, the design in [56] achieves the best generalization performance, and serves as an upper bound. It is seen that the MHA module can effectively enhance the generalization performance of the teacher, increasing the accuracies of ATop-1 and ATop-3 by 2 percentage points. Moreover, with self-KD, the performance of the teacher is significantly improved. For instance, 95% ADBA is achieved with self-KD, which is over 98% of the optimal accuracy. Note that approximately 6.4×10^7 model parameters are required in Jiang’s design (including YOLOv4). In contrast, the teacher model with MHA has only approximately 1.8×10^6 parameters, achieving a reduction in complexity of over 97%.

Table III shows the overall generalization performance of the student model with $L = 8, 5, 3$. We perform a grid search to find the optimal values of β and Γ in the experiments, which are summarized in Table IV. We draw the following observations. First, the generalization performance of the student model generally deteriorates with shorter input sequences. For example, the ATop-3 accuracy of the student (with or without KD) is reduced by 7 percentage points when L decreases from 8 to 3. Such results are reasonable since less input data provides less useful semantic information about the target. Second, it is seen that KD significantly improves the performance of the student model. For example, the student model with $L = 3$ outperforms the “vanilla” student trained from the dataset without KD for $L = 8$. In particular, the former attains 91.56% ADBA and 90.57% ATop-5 accuracy, while the

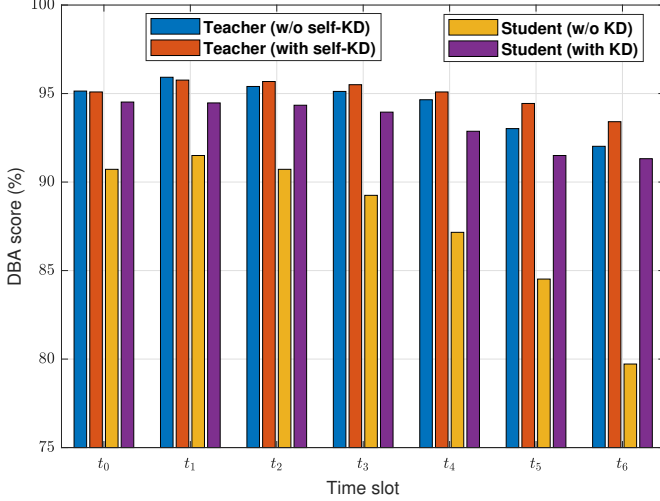
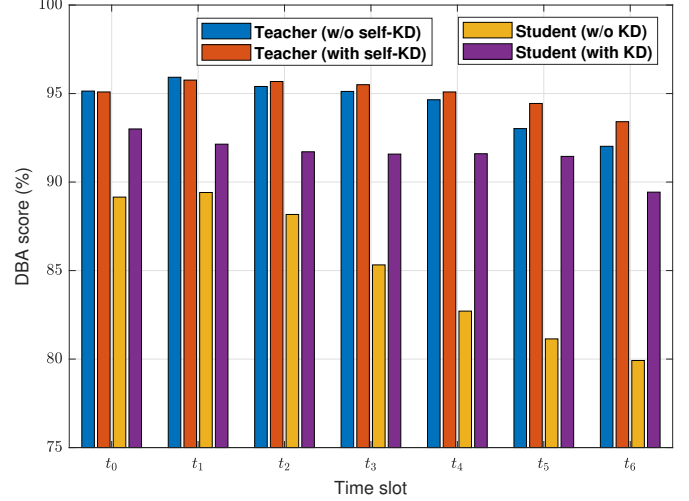
Fig. 7. Performance of the teacher and student models with $L = 8$.Fig. 8. Top-3 and Top-5 prediction accuracy of the student model. The teacher models are trained and tested under $L = 8$.TABLE IV. Values of β and Γ for KD-aided learning.

Term	Teacher (self-KD)	Student		
		$L = 8$	$L = 5$	$L = 3$
β	0.3	0.3	0.5	0.5
Γ	2	5	3	4

corresponding values for the latter are 90.31% and 88.32%, respectively. Third, with the aid of KD, the student model can achieve performance comparable to the vanilla teacher model without self-KD. For example, the student model trained with KD for $L = 8$ achieves 94.86% ADBA, compared with 94.47% for the teacher model without KD. Moreover, the student model trained with KD for $L = 3$ achieves 96% the performance of the self-KD teacher, despite the fact that the former requires only 37.5% of the input data used by the latter, resulting in a complexity reduction of over 90%.

E. Performance for Specific Time Slots

Fig. 7 shows the performance of the teacher and student models with $L = 8$. It is seen that, as expected, the teacher and student achieve lower prediction accuracy and DBA scores for beam prediction farther into the future. Comparing the performance of the teacher with and without MHA in Fig. 7(a), we observe that the integrated attention mechanism primarily improves the prediction accuracy of later time slots, i.e., t_4 – t_6 . Such results verify that the MHA can effectively capture time dependencies inherent in the input sequence, extracting useful information for more challenging future beam predictions. Similarly, the teacher with self-KD mainly improves the performance of beam prediction for time slots farther in the future. With self-KD, the additional soft distribution information between candidate beam classes provides an extra regularizer that helps the model learn better for more challenging tasks [44]. The effectiveness of KD is particularly prominent for the student model, which has significant

(a) Student model with $L = 5$.(b) Student model with $L = 3$.Fig. 9. DBA score of the student model with $L = 3, 5$. The teacher models are trained and tested under $L = 8$.

gaps in prediction accuracy compared to the teacher when implemented without KD-aided learning. In contrast, with KD, the student model achieves close to 95% Top-5 accuracy, which even surpasses the vanilla teacher without self-KD at the future time slots t_5 and t_6 . Although the Top-3 accuracy of the teacher and student models is no more than 85%, the corresponding DBA scores can reach 95%, as seen in Fig. 7(b).

Figs. 8(a) and 8(b) show the Top-3 and Top-5 prediction accuracy of the student models with $L = 5$ and $L = 3$, respectively, where the teacher is trained and tested with $L = 8$ for comparison. The teacher model with/without self-KD specifically refers to the architecture employing MHA. It is observed that KD significantly improves the performance of the student model. For example, percentage-point enhancements of 16.9 and 14.6 in Top-3 accuracy are achieved at time slots t_6 and t_5 for KD-aided learning for $L = 5$ and $L = 3$, respectively. The Top-5 accuracy attained by the student with $L = 5$ and $L = 3$ is improved by 12.8 and 14.5 percentage points at time slots t_6 and t_5 , respectively. Furthermore, the student with KD for $L = 5$ achieves slightly higher Top-5 prediction accuracy at t_6 than the vanilla teacher model with $L = 8$. The student with KD achieves over 90% Top-5 prediction accuracy for both $L = 5$ and $L = 3$, showcasing the effectiveness of KD-aided learning. It is also verified from Figs. 9(a) and 9(b) that the DBA scores, which are based on the Top-3 accuracy, can reach as high as 94% and 93% for $L = 5$ and $L = 3$, respectively.

Figs. 10(a)–10(d) show the Top-1, Top-3, and Top-5 prediction accuracies and DBA scores of the teacher and student models, respectively. We observe that KD is particularly effective for Top-1 accuracy improvements of the student model with $L = 5$ and $L = 3$. For instance, the student model with $L = 5$ achieves a percentage-point gain of 17.5 in Top-1 accuracy at time slot t_6 due to KD, whereas the Top-3 accuracy and DBA score are improved by 16.9 and 11.6 percentage points, respectively. The Top-5 accuracy achieved by the student model with $L = 3$ is improved by 14.1 percentage points with KD. In contrast, the benefit of KD for the student model with $L = 8$ is less significant. At

time slot t_3 , percentage-point gains of 6.1 in Top-1 accuracy, 10.5 in Top-3 accuracy, and 5.1 in DBA score are obtained with KD. The reason for the extra improvement is that Top-1 prediction is more challenging than the other two, especially for shorter input sequences and later time slots. For such challenging tasks, the student model struggles to learn well solely from the dataset. Therefore, the additional guidance from the teacher becomes more helpful in such cases.

Fig. 11 demonstrates the generalization performance of the self-KD-aided teacher model, which is trained for $L = 8$ and tested for $L = 5, 3$. In contrast, the student models for comparison are trained and tested for $L = 5$ and $L = 3$, respectively. It is seen that the DBA score achieved by the teacher is lowest at time slot t_0 and highest at time slot t_6 . The surprising increase in accuracy for farther time slots with shorter inputs likely arises from a training-testing mismatch, where the model struggles early due to reduced input context but benefits from internal GRU decoder dynamics (e.g., autoregressive accumulation of the hidden states) in farther time slots. In contrast, the student with KD maintains relatively high DBA scores across all time slots, verifying its robustness.

VI. CONCLUSIONS

This work has proposed a KD-based vision-assisted long-term beam tracking framework that predicts both current and future beams from past sensor observations. A high-capacity teacher model, built with CNNs, GRUs, and MHA, is first developed to achieve high predictive accuracy. Leveraging KD, a lightweight student model is trained not only to compress the architecture but also to operate effectively with shorter input sequences while preserving long-term prediction ability. Experimental results show that the teacher approaches state-of-the-art long-term prediction accuracies with a 90% reduction in complexity. The student model closely matches the teacher's performance while using less than 10% of its computational resources, and it maintains over 90% long-term beam prediction accuracy even with 60% fewer inputs. These advantages substantially reduce power consumption, sensing and processing

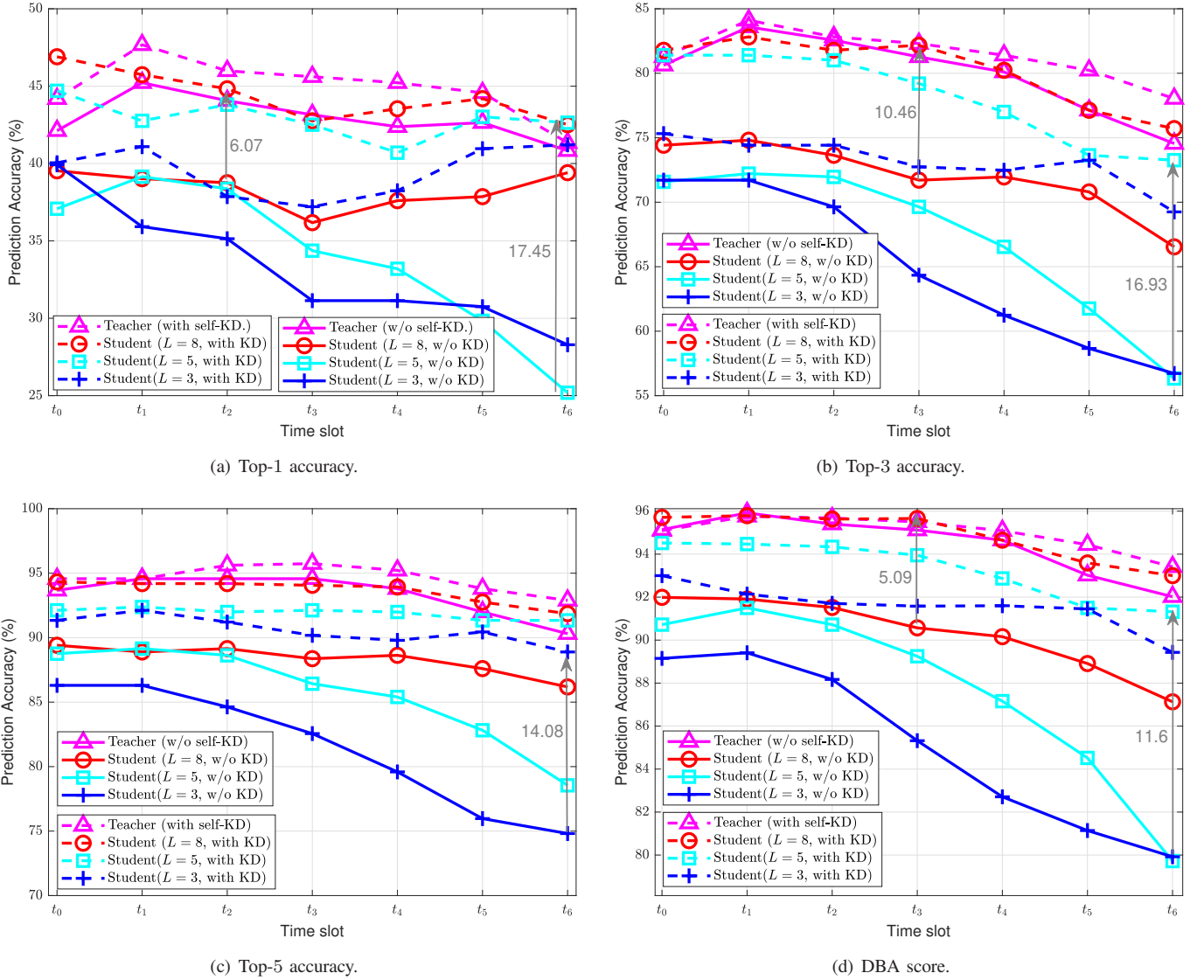


Fig. 10. Performance of the teacher and student models.

latency, and signaling overhead, advancing practical deployment in resource-constrained ISAC systems. The proposed approach provides a viable pathway toward high-accuracy, low-latency, and energy-efficient long-term beam tracking. Future research may extend the framework to incorporate multiple sensing modalities for more challenging real-world scenarios.

REFERENCES

- [1] W. Jiang, B. Han, M. A. Habibi, and H. D. Schotten, "The road towards 6G: A comprehensive survey," *IEEE Open J. Commun. Soc.*, vol. 2, pp. 334–366, 2021.
- [2] W. Yi, W. Zhiqing, and F. Zhiyong, "Beam training and tracking in mmwave communication: A survey," *China Commun.*, vol. 21, no. 6, pp. 1–22, 2024.
- [3] S. Imran, G. Charan, and A. Alkhateeb, "Environment semantic communication: Enabling distributed sensing aided networks," *IEEE Open J. Commun. Soc.*, vol. 5, pp. 7767–7786, 2024.
- [4] X. Cheng, D. Duan, S. Gao, and L. Yang, "Integrated sensing and communications (ISAC) for vehicular communication networks (VCN)," *IEEE Internet Things J.*, vol. 9, no. 23, pp. 23 441–23 451, 2022.
- [5] N. Shlezinger, M. Ma, O. Lavi, N. T. Nguyen, Y. C. Eldar, and M. Juntti, "Artificial intelligence-empowered hybrid multiple-input/multiple-output beamforming: Learning to optimize for high-throughput scalable MIMO," *IEEE Veh. Technol. Mag.*, vol. 19, no. 3, pp. 58–67, 2024.
- [6] M. Ma, T. Fang, N. Shlezinger, A. Swindlehurst, M. Juntti, and N. Nguyen, "Model-based machine learning for Max-Min fairness beamforming design in JCAS systems," in *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Process.*, 2025.
- [7] Z. Xiao, T. He, P. Xia, and X.-G. Xia, "Hierarchical codebook design for beamforming training in millimeter-wave communication," *IEEE Trans. Wireless Commun.*, vol. 15, no. 5, pp. 3380–3392, 2016.
- [8] M. Li, C. Liu, S. V. Hanly, I. B. Collings, and P. Whiting, "Explore and eliminate: Optimized two-stage search for millimeter-wave beam alignment," *IEEE Trans. Wireless Commun.*, vol. 18, no. 9, pp. 4379–4393, 2019.
- [9] C. Qi, K. Chen, O. A. Dobre, and G. Y. Li, "Hierarchical codebook-based multiuser beam training for millimeter wave massive MIMO," *IEEE Trans. Wireless Commun.*, vol. 19, no. 12, pp. 8142–8152, 2020.
- [10] S. Jayaprakasam, X. Ma, J. W. Choi, and S. Kim, "Robust beam-tracking for mmWave mobile communications," *IEEE Commun. Lett.*, vol. 21, no. 12, pp. 2654–2657, 2017.
- [11] J. Lim, H.-M. Park, and D. Hong, "Beam tracking under highly nonlinear mobile millimeter-wave channel," *IEEE Commun. Lett.*, vol. 23, no. 3, pp. 450–453, 2019.
- [12] C. Liu, M. Li, L. Zhao, P. Whiting, S. V. Hanly, I. B. Collings, and M. Zhao, "Robust adaptive beam tracking for mobile millimetre wave communications," *IEEE Trans. Wireless Commun.*, vol. 20, no. 3, pp. 1918–1934, 2020.
- [13] D. Zhang, A. Li, M. Shirvanimoghaddam, P. Cheng, Y. Li, and B. Vucetic, "Codebook-based training beam sequence design for

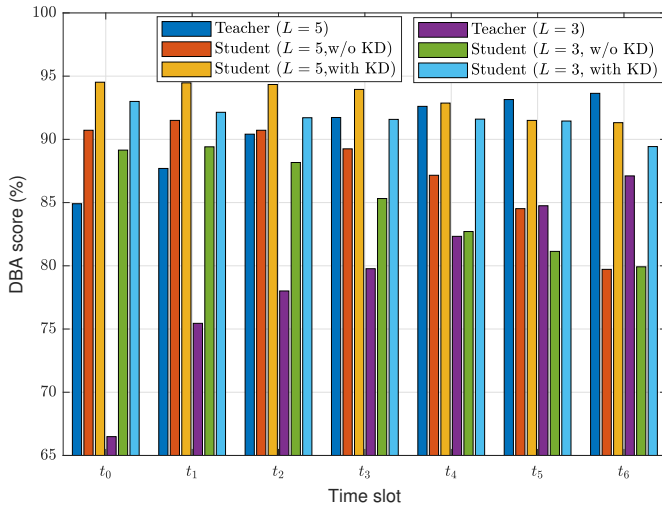


Fig. 11. Generalization performance of the teacher model (with self-KD). The teacher model is trained for $L = 8$ and tested for $L = 5, 3$. In contrast, the student models for comparison are trained and tested for $L = 5$ and $L = 3$, respectively.

millimeter-wave tracking systems,” *IEEE Trans. Wireless Commun.*, vol. 18, no. 11, pp. 5333–5349, 2019.

- [14] D. Zhang, A. Li, H. Chen, N. Wei, M. Ding, Y. Li, and B. Vucetic, “Beam allocation for millimeter-wave MIMO tracking systems,” *IEEE Trans. Veh. Technol.*, vol. 69, no. 2, pp. 1595–1611, 2020.
- [15] M. Ma, N. T. Nguyen, and M. Juntti, “Closed-form hybrid beamforming solution for spectral efficiency upper bound maximization in mmWave MIMO-OFDM systems,” in *Proc. IEEE Veh. Technol. Conf.*, 2021.
- [16] A. Alkhateeb, S. Alex, P. Varkey, Y. Li, Q. Qu, and D. Tujkovic, “Deep learning coordinated beamforming for highly-mobile millimeter wave systems,” *IEEE Access*, vol. 6, pp. 37 328–37 348, 2018.
- [17] L.-H. Shen, T.-W. Chang, K.-T. Feng, and P.-T. Huang, “Design and implementation for deep learning based adjustable beamforming training for millimeter wave communication systems,” *IEEE Trans. Veh. Technol.*, vol. 70, no. 3, pp. 2413–2427, 2021.
- [18] J. Liu, X. Li, T. Fan, S. Lv, and M. Shi, “Model-driven deep learning assisted beam tracking for millimeter-wave systems,” *IEEE Commun. Lett.*, vol. 26, no. 10, pp. 2345–2349, 2022.
- [19] M. Fozi, A. R. Sharafat, and M. Bennis, “Fast MIMO beamforming via deep reinforcement learning for high mobility mmWave connectivity,” *IEEE J. Sel. Areas Commun.*, vol. 40, no. 1, pp. 127–142, 2022.
- [20] K. Ma, F. Zhang, W. Tian, and Z. Wang, “Continuous-time mmWave beam prediction with ODE-LSTM learning architecture,” *IEEE Wireless Commun. Lett.*, vol. 12, no. 1, pp. 187–191, 2023.
- [21] N. T. Nguyen, M. Ma, O. Lavi, N. Shlezinger, Y. C. Eldar, A. L. Swindlehurst, and M. Juntti, “Deep unfolding hybrid beamforming designs for THz massive MIMO systems,” *IEEE Trans. Signal Process.*, vol. 71, pp. 3788–3804, 2023.
- [22] J. Liu, X. Li, T. Fan, S. Lv, and M. Shi, “Multimodal fusion assisted mmwave beam training in dual-model networks,” *IEEE Trans. Veh. Technol.*, vol. 73, no. 1, pp. 995–1011, 2024.
- [23] J. Mu, Y. Gong, F. Zhang, Y. Cui, F. Zheng, and X. Jing, “Integrated sensing and communication-enabled predictive beamforming with deep learning in vehicular networks,” *IEEE Commun. Lett.*, vol. 25, no. 10, pp. 3301–3304, 2021.
- [24] C. Liu, W. Yuan, S. Li, X. Liu, H. Li, D. W. K. Ng, and Y. Li, “Learning-based predictive beamforming for integrated sensing and communication in vehicular networks,” *IEEE J. Sel. Areas Commun.*, vol. 40, no. 8, pp. 2317–2334, 2022.
- [25] S. Jiang, G. Charan, and A. Alkhateeb, “LiDAR aided future beam prediction in real-world millimeter wave V2I communications,” *IEEE Wireless Commun. Lett.*, vol. 12, no. 2, pp. 212–216, 2023.
- [26] A. Klautau, N. González-Precicic, and R. W. Heath, “LiDAR data for deep learning-based mmWave beam-selection,” *IEEE Wireless Commun. Lett.*, vol. 8, no. 3, pp. 909–912, 2019.
- [27] U. Demirhan and A. Alkhateeb, “Radar aided 6G beam prediction: Deep learning algorithms and real-world demonstration,” in *Proc. IEEE Wireless Commun. and Networking Conf.*, 2022.
- [28] H. Luo, U. Demirhan, and A. Alkhateeb, “Millimeter wave V2V beam tracking using radar: Algorithms and real-world demonstration,” in *Proc. European Signal Proc. Conf.*, 2023.
- [29] Y. Yang, F. Gao, X. Tao, G. Liu, and C. Pan, “Environment semantics aided wireless communications: A case study of mmWave beam prediction and blockage prediction,” *IEEE J. Sel. Areas Commun.*, vol. 41, no. 7, pp. 2025–2040, 2023.
- [30] G. Charan, T. Osman, A. Hredzak, N. Thawdar, and A. Alkhateeb, “Vision-position multi-modal beam prediction using real millimeter wave datasets,” in *Proc. IEEE Wireless Commun. and Networking Conf.*, 2022.
- [31] W. Xu, F. Gao, S. Jin, and A. Alkhateeb, “3D scene-based beam selection for mmWave communications,” *IEEE Wireless Commun. Lett.*, vol. 9, no. 11, pp. 1850–1854, 2020.
- [32] X. Cheng, H. Zhang, J. Zhang, S. Gao, S. Li, Z. Huang, L. Bai, Z. Yang, X. Zheng, and L. Yang, “Intelligent multi-modal sensing-communication integration: Synesthesia of machines,” *IEEE Commun. Surveys Tuts.*, vol. 26, no. 1, pp. 258–301, 2024.
- [33] Y. Cui, J. Nie, X. Cao, T. Yu, J. Zou, J. Mu, and X. Jing, “Sensing-assisted high reliable communication: A transformer-based beamforming approach,” *IEEE J. Sel. Topics Signal Process.*, vol. 18, no. 5, pp. 782–795, 2024.
- [34] S. Tariq, B. E. Arfeto, U. Khalid, S. Kim, T. Q. Duong, and H. Shin, “Deep quantum-transformer networks for multimodal beam prediction in ISAC systems,” *IEEE Internet Things J.*, vol. 11, no. 18, pp. 29 387–29 401, 2024.
- [35] Y. Tian, Q. Zhao, F. Boukhalfa, K. Wu, F. Bader *et al.*, “Multimodal transformers for wireless communications: A case study in beam prediction,” *arXiv preprint arXiv:2309.11811*, 2023.
- [36] B. Shi, M. Li, M.-M. Zhao, M. Lei, and L. Li, “Multimodal deep learning empowered millimeter-wave beam prediction,” in *Proc. IEEE Veh. Technol. Conf.*, 2024.
- [37] Y. M. Park, Y. K. Tun, W. Saad, and C. S. Hong, “Resource-efficient beam prediction in mmwave communications with multimodal realistic simulation framework,” *arXiv preprint arXiv:2504.05187*, 2025.
- [38] K. Zhang, W. Yu, H. He, S. Song, J. Zhang, and K. B. Letaief, “Multimodal deep learning-empowered beam prediction in future THz ISAC systems,” *arXiv preprint arXiv:2505.02381*, 2025.
- [39] Q. Zhu, Y. Wang, W. Li, H. Huang, and G. Gui, “Advancing multi-modal beam prediction with cross-modal feature enhancement and dynamic fusion mechanism,” *IEEE Trans. Commun.*, Early access, 2025.
- [40] G. Hinton, O. Vinyals, and J. Dean, “Distilling the knowledge in a neural network,” *arXiv preprint arXiv:1503.02531*, 2015.
- [41] K. Zhang, H. Ying, H.-N. Dai, L. Li, Y. Peng, K. Guo, and H. Yu, “Compacting deep neural networks for Internet of Things: Methods and applications,” *IEEE Internet Things J.*, vol. 8, no. 15, pp. 11 935–11 959, 2021.
- [42] M. Phuong and C. Lampert, “Towards understanding knowledge distillation,” in *Proc. International Conference on Machine Learning*, PMLR, 2019, pp. 5142–5151.
- [43] J. Tang, R. Shivanna, Z. Zhao, D. Lin, A. Singh, E. H. Chi, and S. Jain, “Understanding and improving knowledge distillation,” *arXiv preprint arXiv:2002.03532*, 2020.
- [44] H. Mobahi, M. Farajtabar, and P. Bartlett, “Self-distillation amplifies regularization in Hilbert space,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 3351–3361, 2020.
- [45] C. Bucilua, R. Caruana, and A. Niculescu-Mizil, “Model compression,” in *Proc. 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2006, pp. 535–541.
- [46] J. Ba and R. Caruana, “Do deep nets really need to be deep?” *Advances in Neural Information Processing Systems*, vol. 27, 2014.
- [47] A. Polino, R. Pascanu, and D. Alistarh, “Model compression via distillation and quantization,” *arXiv preprint arXiv:1802.05668*, 2018.
- [48] K. Kong, W.-J. Song, and M. Min, “Knowledge distillation-aided end-to-end learning for linear precoding in multiuser MIMO downlink systems with finite-rate feedback,” *IEEE Trans. Veh. Technol.*, vol. 70, no. 10, pp. 11 095–11 100, 2021.
- [49] H. Tang, J. Guo, M. Matthaiou, C.-K. Wen, and S. Jin, “Knowledge-distillation-aided lightweight neural network for massive MIMO CSI feedback,” in *Proc. IEEE Veh. Technol. Conf.*, 2021.
- [50] F. O. Catak, M. Kuzlu, E. Catak, U. Cali, and O. Guler, “Defensive distillation-based adversarial attack mitigation method for channel estimation using deep learning models in next-generation wireless networks,” *IEEE Access*, vol. 10, pp. 98 191–98 203, 2022.
- [51] J. Guo, C.-K. Wen, M. Chen, and S. Jin, “Environment knowledge-aided massive MIMO feedback codebook enhancement using artificial intelligence,” *IEEE Trans. Commun.*, vol. 70, no. 7, pp. 4527–4542, 2022.

- [52] C. Liu, Y. Zhou, Y. Chen, and S.-H. Yang, "Knowledge distillation-based semantic communications for multiple users," *IEEE Trans. Wireless Commun.*, vol. 23, no. 7, pp. 7000–7012, 2023.
- [53] A. Al-Ahmadi, "Knowledge distillation based deep learning model for user equipment positioning in massive MIMO systems using flying reconfigurable intelligent surfaces," *IEEE Access*, vol. 12, pp. 20 679–20 691, 2024.
- [54] Y. Zhang, Z. Yan, X. Sun, W. Diao, K. Fu, and L. Wang, "Learning efficient and accurate detectors with dynamic knowledge distillation in remote sensing imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–19, 2021.
- [55] J. Ni, R. Sarbajna, Y. Liu, A. H. Ngu, and Y. Yan, "Cross-modal knowledge distillation for vision-to-sensor action recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Process.*, 2022.
- [56] S. Jiang and A. Alkhateeb, "Computer vision aided beam tracking in a real-world millimeter wave deployment," in *Proc. IEEE Global Commun. Conf. Workshop*, 2022.
- [57] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020.
- [58] M. Thomas and A. T. Joy, *Elements of Information Theory*. Wiley-Interscience, 2006.
- [59] A. Sherstinsky, "Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network," *Physica D: Nonlinear Phenomena*, vol. 404, p. 132306, 2020.
- [60] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.
- [61] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altschmidt, S. Altman, S. Anadkat *et al.*, "GPT-4 Technical Report," *arXiv preprint arXiv:2303.08774*, 2023.
- [62] A. Liu, B. Feng, B. Xue, B. Wang, B. Wu, C. Lu, C. Zhao, C. Deng, C. Zhang, C. Ruan *et al.*, "Deepseek-v3 Technical Report," *arXiv preprint arXiv:2412.19437*, 2024.
- [63] R. Dey and F. M. Salem, "Gate-variants of gated recurrent unit (GRU) neural networks," in *Proc. IEEE International Midwest Symp. on Circuits and Systems (MWSCAS)*, 2017, pp. 1597–1600.
- [64] Z. Liu, M. Sun, T. Zhou, G. Huang, and T. Darrell, "Rethinking the value of network pruning," *arXiv preprint arXiv:1810.05270*, 2018.
- [65] B. Rokh, A. Azarpeyvand, and A. Khanteymoori, "A comprehensive survey on model quantization for deep neural networks in image classification," *ACM Transactions on Intelligent Systems and Technology*, vol. 14, no. 6, pp. 1–50, 2023.
- [66] S. Stanton, P. Izmailov, P. Kirichenko, A. A. Alemi, and A. G. Wilson, "Does knowledge distillation really work?" *Advances in Neural Information Processing Systems*, vol. 34, pp. 6906–6919, 2021.
- [67] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE International Conf. on Computer Vision*, 2017, pp. 2980–2988.
- [68] A. Alkhateeb, G. Charan, T. Osman, A. Hredzak, J. Morais, U. Demirhan, and N. Srinivas, "Deepsense 6G: A large-scale real-world multi-modal sensing and communication dataset," *IEEE Commun. Mag.*, vol. 61, no. 9, pp. 122–128, 2023.
- [69] G. Charan, U. Demirhan, J. Morais, A. Behboodi, H. Pezeshki, and A. Alkhateeb, "Multi-modal beam prediction challenge 2022: Towards generalization," *arXiv preprint arXiv:2209.07519*, 2022.