

Machine Learning for Beam Alignment in Millimeter Wave Massive MIMO

Wenyan Ma, *Student Member, IEEE*, Chenhao Qi, *Senior Member, IEEE*,
and Geoffrey Ye Li, *Fellow, IEEE*

Abstract—This article investigates beam alignment for multi-user millimeter wave (mmWave) massive multi-input multi-output system. Unlike the existing works using machine learning (ML), an alignment method with partial beams using ML (AMPBML) is proposed without any prior knowledge such as user location information. The neural network (NN) for the AMPBML is trained offline using simulated environments according to the mmWave channel model and is then deployed online to predict the beam distribution vector using partial beams. Afterwards, the beams for all users are all aligned simultaneously based on the indices of the dominant entries of the obtained beam distribution vector. Simulation results demonstrate that the AMPBML outperforms the existing methods, including the adaptive compressed sensing, hierarchical search, and multi-path decomposition and recovery, in terms of the total training time slots and the spectral efficiency.

Index Terms—Beam alignment, machine learning, massive MIMO, millimeter wave communications.

I. INTRODUCTION

Due to its abundant frequency spectrum resource, millimeter wave (mmWave) communications have attracted broad attention and become an important technology in the future [1]. The mmWave signal experiences high path loss but can be compensated by utilizing a massive multi-input multi-output (MIMO) antenna array to achieve directional beam alignment (BA) and data transmission. Its short wavelength enables large antenna arrays to be packed into small form factors.

In multi-user multi-stream mmWave systems, the base station (BS) simultaneously serves multiple users with multiple beams. To align beams for different users, hierarchical codebook based beam training is usually used. For examples, an adaptive compressed sensing (ACS) method has been proposed in [1], where a hierarchical codebook with multi-resolution is designed to train beams for all users sequentially. Then a hierarchical search (HS) method, which can be performed much faster than the ACS method, has been developed in [2]. To combine the advantages of the HS and the ACS, a multi-path decomposition and recovery (MDR) method has been proposed in [3]. However, using hierarchical codebook based

beam training to align beams for multiple users is not trivial. The BS has to align beams for all users sequentially during the training stage, leading to huge overhead. Moreover, the optimal codeword index must be fed back for each layer of the hierarchical codebook, which is also time consuming. Other work investigates the beam alignment in mmWave systems equipped with automotive sensors or radars, where the BS can obtain the user location information from these sensors or radars and the directional beams can be designed. For examples, a beam alignment solution is designed by extracting useful information from radar signal [4], while the beams are aligned based on the location information from automotive sensors [5]. However, the equipped automotive sensors or radars will incur additional hardware overhead.

Recently, machine learning (ML) has been applied to address different issues in physical layer communications [6]. The application of ML to BA in mmWave systems has also been investigated to take the advantages of ML in solving complicated nonlinear problems. For examples, the ML tools and situational awareness are combined to learn the beam information including power and optimal beam index [7] while the angles of arrival (AoAs) can be estimated and input to the neural network (NN) for beam selection [8]. However, the above two ML methods need the location information of the users to train the NN, which incurs extra system overhead.

In this article, we propose an alignment method with partial beams using ML (AMPBML) for the multi-user mmWave massive MIMO system. The NN for the AMPBML is trained offline using simulated environments according to the mmWave channel model and is then deployed online to predict the beam distribution vector using partial beams. Afterwards, the beams for all users are aligned simultaneously based on the obtained indices of the dominant entries of beam distribution vector. Different from the existing works based on hierarchical codebook, we align beams for all users simultaneously and significantly save the total training time. Moreover, unlike the existing works on BA using ML, we need no prior knowledge, such as the user location information, to train the NN. It will reduce the system overhead significantly.

Notations: Symbols for vectors (lower case) and matrices (upper case) are in boldface. $(\cdot)^T$, $(\cdot)^*$, $(\cdot)^H$, and $(\cdot)^{-1}$ denote the transpose, conjugate, conjugate transpose, and inverse, respectively. We use \mathbf{I}_L to represent identity matrix of size L . The set of $M \times N$ complex-valued matrices and integral-valued matrices is denoted as $\mathbb{C}^{M \times N}$ and $\mathbb{Z}^{M \times N}$, respectively. We use $\mathbb{E}\{\cdot\}$ to denote expectation. $\|\cdot\|_2$ and $\|\cdot\|_F$ denote l_2 -norm of a vector and Frobenius norm of a matrix, respectively.

This work was supported in part by National Natural Science Foundation of China under Grant 61871119, by Natural Science Foundation of Jiangsu Province under Grant BK20161428, and by the Fundamental Research Funds for the Central Universities. (*Corresponding author: Chenhao Qi*)

Wenyan Ma and Chenhao Qi are with the School of Information Science and Engineering, Southeast University, Nanjing 210096, China (Email: qch@seu.edu.cn).

Geoffrey Ye Li is with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA (Email: liye@ece.gatech.edu).

The n th entry of \mathbf{a} is denoted as $\mathbf{a}[n]$. We use $\lfloor \cdot \rfloor$ and $\lceil \cdot \rceil$ to denote floor and ceil operations, respectively. Complex Gaussian distribution is denoted as \mathcal{CN} .

II. SYSTEM MODEL AND PROBLEM FORMULATION

We first introduce the multi-user mmWave massive MIMO system. Then we formulate the BA problem for multiple users.

A. System Model

We consider an uplink multi-user mmWave massive MIMO communication system comprising a BS and U users. The BS is equipped with a uniform linear array (ULA) [1]. Note that the present method can be generalized to other array structures. Hybrid combining is typically adopted, where the number of antennas, N_A , is much larger than that of RF chains, N_R , i.e., $N_A \gg N_R$.

For uplink transmission, hybrid combining at the BS consists of baseband digital combining and RF analog combining [9]. Denote s_u to be the transmit signal. Then the received signal vector at the BS can be expressed as

$$\mathbf{y} = \mathbf{W}_B^H \mathbf{W}_R^H \sum_{u=1}^U \mathbf{h}_u s_u + \mathbf{W}_B^H \mathbf{W}_R^H \mathbf{n}, \quad (1)$$

where $\mathbf{W}_B \in \mathbb{C}^{N_R \times N_R}$ and $\mathbf{W}_R \in \mathbb{C}^{N_A \times N_R}$ are the digital combining matrix and analog combining matrix, respectively, and $\mathbf{n} \in \mathbb{C}^{N_A}$ is the additive white Gaussian noise (AWGN) vector satisfying $\mathbf{n} \sim \mathcal{CN}(0, \sigma^2 \mathbf{I}_{N_A})$. To normalize the power of the hybrid combiner, we set $\|\mathbf{W}_B^H \mathbf{W}_R^H\|_F^2 = N_R$ and $\mathbb{E}\{s_u s_u^*\} = 1$.

There are different kinds of channel model in mmWave systems, such as the clustered mmWave channel model [10], [11] and the Saleh-Valenzuela mmWave channel model [1]. We choose the Saleh-Valenzuela mmWave channel model in our paper. For the Saleh-Valenzuela mmWave channel model [1], the channel vector $\mathbf{h}_u \in \mathbb{C}^{N_A}$ between the u th user and the BS can be represented by

$$\mathbf{h}_u = \sqrt{\frac{N_A}{L_u}} \sum_{i=1}^{L_u} \mathbf{h}_{u,i} = \sqrt{\frac{N_A}{L_u}} \sum_{i=1}^{L_u} g_{u,i} \boldsymbol{\alpha}(N_A, \theta_{u,i}), \quad (2)$$

where $\mathbf{h}_{u,i}$, L_u , and $g_{u,i}$ are denoted as the channel vector, number of multiple channel paths, and complex gain of the i th path, respectively, and the steering vector $\boldsymbol{\alpha}(N, \theta)$ in (2) can be expressed as $\boldsymbol{\alpha}(N, \theta) = \frac{1}{\sqrt{N}} [1, e^{j\pi\theta}, \dots, e^{j\pi\theta(N-1)}]^T$ [2], [3]. Typically \mathbf{h}_u consists of one line-of-sight (LOS) path (the 1st channel path), and $L_u - 1$ non-line-of-sight (NLOS) paths (the i th channel path for $2 \leq i \leq L_u$).

The AoA of the i th path of the u th user $\vartheta_{u,i}$ is uniformly distributed over $[-\pi, \pi]$ [1]. Then $\theta_{u,i} \triangleq \sin \vartheta_{u,i}$ in (2) if the distance between adjacent antennas at the BS is with half-wave length.

B. Problem Formulation

According to [9], each column of the analog combiner is a codeword selected from a beam steering codebook, which consists of N_A equally spaced channel steering vectors pointing

at N_A different directions. The codebook at the BS is denoted by $\mathcal{W} \triangleq \{\mathbf{w}(1), \mathbf{w}(2), \dots, \mathbf{w}(N_A)\}$, where

$$\mathbf{w}(n) = \boldsymbol{\alpha}(N_A, -1 + (2n - 1)/N_A). \quad (3)$$

Our objective is to search U beams from the total N_A ones that align with the LOS paths of U users. Denote $\mathbf{C} \triangleq [\mathbf{w}(1), \mathbf{w}(2), \dots, \mathbf{w}(N_A)] \in \mathbb{C}^{N_A \times N_A}$ and $\boldsymbol{\beta}_u \triangleq \mathbf{C}^H \boldsymbol{\alpha}(N_A, \theta_{u,1}) \in \mathbb{C}^{N_A}$. Then the objective can be denoted as

$$\arg \max_{n=1,2,\dots,N_A} |\boldsymbol{\beta}_u[n]|, \quad u = 1, 2, \dots, U. \quad (4)$$

Denote $\mathbf{b} \triangleq [b_1, b_2, \dots, b_U]^T \in \mathbb{Z}^U$ as the U beam indices that align with the LOS paths of U users, where b_u can be denoted as

$$b_u = \left\lfloor \frac{N_A(\theta_{u,1} + 1)}{2} \right\rfloor + 1 \in \{1, 2, \dots, N_A\}. \quad (5)$$

Given $\theta_{u,1}$, the objective U beam indices can be directly calculated according to (5). However, $\theta_{u,1}$ is unknown before channel training. Exhaustive search to solve (4) for the best analog combiner requires $N_A U / N_R$ time slots. For example, $N_A U / N_R = 256$ if $N_A = 256$, $U = 3$, and $N_R = 3$ [9]. The exhaustive search requiring the full knowledge of $\boldsymbol{\beta}_u$ is rather time consuming if there are a large number of antennas. In order to reduce the time slots, we use only part of total N_A beams to predict the optimal U beams. However, due to the nonlinearity of $\boldsymbol{\beta}_u$, it is hard to directly calculate the optimal beam based on partial entries of $\boldsymbol{\beta}_u$. Recently, ML has shown advantages on dealing with nonlinear problems. Therefore we will develop ML based BA based on partial entries of $\boldsymbol{\beta}_u$.

III. AMPBML

The major steps of the AMPBML is summarized in Algorithm 1. During the uplink beam training, we use J different analog combining matrices, denoted as $\mathbf{W}_R^t \in \mathbb{C}^{N_A \times N_R}$. The beam alignment chooses analog beams that best align with the LOS path of each user, which is independent of the digital combining matrix. Therefore we set $\mathbf{W}_B = \mathbf{I}_{N_R}$ at the BS. For simplicity, each user transmits the same signal $s_u = 1$ for all J slots. The channel is assumed to be time-invariant during J time slots. The received signal vector at the BS at the t th slot can be denoted as

$$\mathbf{y}^t = (\mathbf{W}_R^t)^H \sum_{u=1}^U \mathbf{h}_u + \tilde{\mathbf{n}}^t, \quad (6)$$

where $\tilde{\mathbf{n}}^t \triangleq (\mathbf{W}_R^t)^H \mathbf{n}^t \in \mathbb{C}^{N_R}$ and $\mathbf{n}^t \in \mathbb{C}^{N_A}$ is the AWGN vector, each entry of which is with independent complex Gaussian distribution with zero mean and variance of σ^2 .

Denote $M \triangleq J N_R$. We stack the J received signal vectors together and have

$$\mathbf{r} = (\mathbf{W})^H \sum_{u=1}^U \mathbf{h}_u + \tilde{\mathbf{n}}, \quad (7)$$

where

$$\begin{aligned} \mathbf{r} &\triangleq [(\mathbf{y}^1)^T, (\mathbf{y}^2)^T, \dots, (\mathbf{y}^J)^T]^T \in \mathbb{C}^M, \\ \mathbf{W} &\triangleq [\mathbf{W}_R^1, \mathbf{W}_R^2, \dots, \mathbf{W}_R^J] \in \mathbb{C}^{N_A \times M}, \\ \tilde{\mathbf{n}} &\triangleq [(\tilde{\mathbf{n}}^1)^T, (\tilde{\mathbf{n}}^2)^T, \dots, (\tilde{\mathbf{n}}^J)^T]^T \in \mathbb{C}^M. \end{aligned} \quad (8)$$

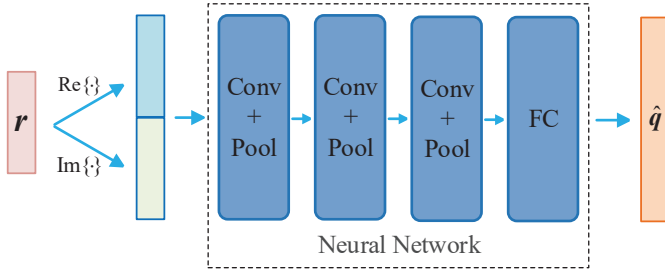


Fig. 1. Illustration of the NN.

Algorithm 1 AMPBML

-
- 1: *Input:* \mathbf{y}^t, J .
 - 2: Obtain \mathbf{r} via (8).
 - 3: Input \mathbf{r} to the offline-trained NN to get $\hat{\mathbf{q}}$.
 - 4: Obtain \mathbf{v} by sorting the absolute value of $\hat{\mathbf{q}}$ in descending order.
 - 5: Calculate $\hat{\mathbf{b}}$ based on (11).
 - 6: *Output:* $\hat{\mathbf{b}}$.
-

To obtain partial entries of β_u , \mathbf{W} is set as a submatrix consisted of M columns of \mathbf{C} . Since \mathbf{r} is determined by \mathbf{h}_u as shown in (7) while \mathbf{h}_u is related to the AoA of the LOS path $\vartheta_{u,1}$ as shown in (2), \mathbf{r} contains the information of the AoA of the LOS path, indicating that we do not require other information except \mathbf{r} to train the NN.

The BA for multiple users has two stages, the offline training of the NN and online deployment. The NN is first trained offline and then used to predict the U beam indices. As in Fig. 1, the input of the NN is \mathbf{r} . The beam distribution vector, denoted by $\mathbf{q} \in \mathbb{Z}^{N_A}$, can be represented as

$$\mathbf{q}[n] = \begin{cases} 1, & n = b_u, u = 1, 2, \dots, U, \\ 0, & \text{otherwise.} \end{cases} \quad (9)$$

The indices of the nonzero entries of \mathbf{q} represent the targeted U beam indices. Note that the AoA is known only during the offline training stage while it is unknown during the online deployment stage. The output of the NN is denoted by $\hat{\mathbf{q}}$ and is expected to be as close to \mathbf{q} as possible.

As illustrated in Fig. 1, the adopted NN in this work consists of three hidden layers and a fully connected (FC) layer. Since the NN can only deal with the real number, the input of the NN is a real-valued vector with the length of $2M$ composed by the imaginary and real parts of \mathbf{r} . Each hidden layer consists of a convolutional (Conv) layer and a pooling (Pool) layer. We use two NN parameter sets: NN parameter set (NPS) I and NPS II. The strides of each Conv layer is set to be 1. The kernel size of each Conv layer is set to be 5 and 10 in NPS I and NPS II, respectively. The numbers of filters of these three Conv layers are set as 16, 32, and 64, respectively, in NPS I and 32, 64, and 128, respectively, in NPS II. The activation function of the Conv layer is the ReLU function, that is $f_{\text{Re}}(x) = \max(0, x)$. Both the pool size and strides of each Pool layer are set to be 2.

During the offline training of the NN, we generate the dataset of \mathbf{r} and \mathbf{q} based on the simulated mmWave channel

environment. With the beam distribution vector in (9) and the received signals in (7), the training data of \mathbf{r} and \mathbf{q} can be obtained. In fact, the process to obtain \mathbf{r} and \mathbf{q} involves the following four steps.

- i) Randomly generate a channel vector based on the mmWave channel model in (2);
- ii) obtain b_u based on (5);
- iii) enumerate the beam distribution vector \mathbf{q} based on (9);
- iv) compute the received signal vector \mathbf{r} based on (7).

We divide the data set into the training set and the validation set randomly, where the size of the training set is nine times the size of the validation set. The output of the NN is $\hat{\mathbf{q}}$.

The training of the NN aims to minimize the difference between $\hat{\mathbf{q}}$ and \mathbf{q} , called the loss in ML. It can be calculated in several ways. In our work, we formulate the beam alignment problem as a multi-label classification problem and use the cross-entropy loss with the sigmoid function as [7]

$$f_{\text{Loss}}(\mathbf{q}, \hat{\mathbf{q}}) = \frac{1}{N_A} \sum_{n=1}^{N_A} \max(\hat{\mathbf{q}}[n], 0) - \hat{\mathbf{q}}[n]\mathbf{q}[n] + \ln(1 + e^{-|\hat{\mathbf{q}}[n]|}). \quad (10)$$

The adaptive moment estimation (Adam) optimizer is used to train the NN by TensorFlow. The NN is trained for 6,000 epochs, where 500 mini-batches are utilized in each epoch. The learning rate is set to be a step function and decreases with the increasing of training epochs. The learning rate is initialized with the value of 0.01 and decreases 5-fold every 1,000 epochs.

During the online deployment of the NN, we obtain the real measured \mathbf{r} from practical mmWave channel environments, which is then input to the offline-trained NN. The prediction of \mathbf{q} by the NN is $\hat{\mathbf{q}}$. Due to the mismatching channel environment between the offline training stage and the online deployment stage, $\hat{\mathbf{q}}$ may not have only U nonzero entries. Therefore, we select U dominant entries of $\hat{\mathbf{q}}$ as the prediction of the U beam indices that align with the LOS paths of U users. Denote $\hat{\mathbf{b}} \triangleq [\hat{b}_1, \hat{b}_2, \dots, \hat{b}_U]^T \in \mathbb{Z}^U$ as the prediction of the U beam indices, which can be represented as

$$\hat{b}_u = \mathbf{v}[u], \quad (11)$$

for $u = 1, 2, \dots, U$.

IV. SIMULATION RESULTS

We consider an mmWave massive MIMO system with a BS equipped with $N_A = 256$ antennas and $N_R = 8$ RF chains. The number of resolvable paths in the mmWave channel is set to be $L_u = 3$, while $g_{u,1} \sim \mathcal{CN}(0, 1)$ and $g_{u,i} \sim \mathcal{CN}(0, 0.01)$ for $i = 2, 3$ [9]. We use $J = 16$ time slots to transmit signals for uplink channel estimation. Then we have $M = JN_R = 128$. \mathbf{W} is set as a submatrix consisted of M columns from \mathbf{C} , with column indices denoted as $1, 1 + \lfloor N_A/M \rfloor, \dots, 1 + (M-1)\lfloor N_A/M \rfloor$ from \mathbf{C} . We compare the proposed AMPBML with the ACS [1], HS [2], and MDR [3] beam training methods.

As shown in Fig. 2, we verify the convergence of the training of the NN. Suppose there are $U = 3, 4, 5$ users, respectively. It is seen that the loss in (10) decreases rapidly

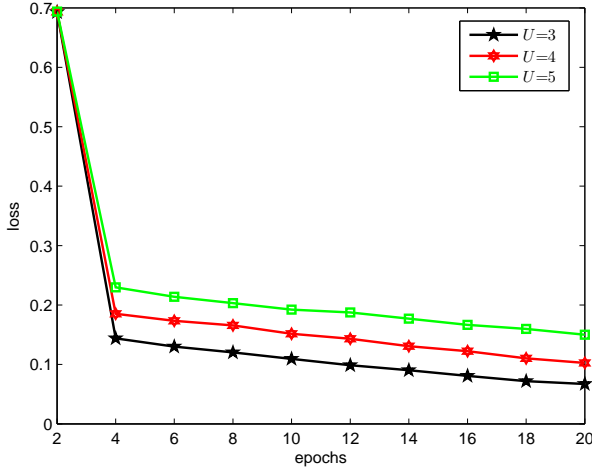


Fig. 2. Convergence of the training of the NN.

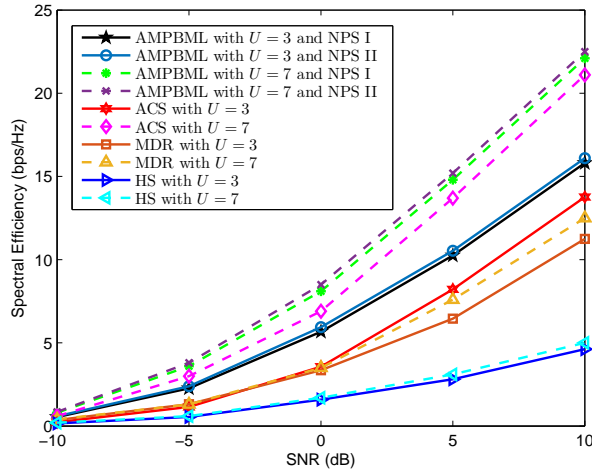


Fig. 3. Comparisons of spectral efficiency for different SNR.

as the number of epochs grows, and the less users lead to the smaller loss. For $U = 3$, the loss is smaller than 0.1 when the number of epochs is larger than 20, which means 20 epochs is enough to achieve the loss smaller than 0.1.

In Fig. 3, we compare the spectral efficiency for different BA methods. During the downlink data transmission, the best analog beamforming vector for the u th user is $\mathbf{w}(\hat{b}_u)$. We use U RF chains to design the analog precoder at the BS as $\tilde{\mathbf{W}}_R = [\mathbf{w}(\hat{b}_1), \mathbf{w}(\hat{b}_2), \dots, \mathbf{w}(\hat{b}_U)]$ [9]. Then the digital precoder is designed as $\tilde{\mathbf{W}}_B = (\tilde{\mathbf{W}}_R^H \tilde{\mathbf{W}}_R)^{-1}$ to eliminate multi-user interference [9]. In order to satisfy the total power constraint, each column of the designed digital precoder, denoted as $\tilde{\mathbf{w}}_{B,u}$, should be normalized, i.e., $\tilde{\mathbf{w}}_{B,u} = \tilde{\mathbf{w}}_{B,u} / \|\tilde{\mathbf{w}}_{B,u}^H \tilde{\mathbf{W}}_R^H\|_2$, such that $\|\tilde{\mathbf{w}}_{B,u}^H \tilde{\mathbf{W}}_R^H\|_2^2 = 1$, $u = 1, 2, \dots, U$. Moreover, since the U beam indices that align with the LOS paths of U users are predicted simultaneously, the user identifier should be transmitted to the BS to determine the one-to-one correspondence of the predicted U

beam indices and U users after the hybrid precoder design. Then the spectral efficiency is denoted as [9]

$$R = \sum_{u=1}^U \log_2 \left(1 + \frac{\frac{1}{U} |\tilde{\mathbf{w}}_{B,u}^H \tilde{\mathbf{W}}_R^H \mathbf{h}_u|^2}{\frac{1}{U} \sum_{i \neq u} |\tilde{\mathbf{w}}_{B,i}^H \tilde{\mathbf{W}}_R^H \mathbf{h}_u|^2 + \sigma^2} \right). \quad (12)$$

Fig. 3 shows that the proposed method achieves better performance than the others when $U = 3$ and $U = 7$. At SNR = 0 dB, the AMPBML with $U = 3$ and NPS I has 59.0%, 68.7%, and 257.0% performance improvement compared with the ACS, MDR, and HS, respectively; while the AMPBML with $U = 7$ and NPS I has 17.3%, 131.4%, and 376.5% performance improvement compared with the ACS, MDR, and HS, respectively. The reason for the better performance of the AMPBML is that it can train the NN with the received signal containing channel noise and is more effective at low SNR. Moreover, the AMPBML with NPS II has slight performance improvement over that with NPS I, since NPS II uses larger kernel size and more filters in each Conv layer.

V. CONCLUSIONS

In this article, we have proposed an AMPBML for multi-user mmWave massive MIMO systems. Simulation results have verified the effectiveness of our work. The proposed method can be used in mmWave communication systems to align beams for multiple users simultaneously to reduce the training slots.

REFERENCES

- [1] A. Alkhateeb, O. El Ayach, G. Leus, and R. W. Heath, "Channel estimation and hybrid precoding for millimeter wave cellular systems," *IEEE J. Sel. Top. Signal Process.*, vol. 8, no. 5, pp. 831–846, Oct. 2014.
- [2] 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, May 2016.
- [3] Z. Xiao, H. Dong, L. Bai, P. Xia, and X.-G. Xia, "Enhanced channel estimation and codebook design for millimeter-wave communication," *IEEE Trans. Veh. Technol.*, vol. 67, no. 10, pp. 9393–9405, Oct. 2018.
- [4] N. Gonzalez-Prelcic, R. Mendez-Rial, and R. W. Heath, "Radar aided beam alignment in mmWave V2I communications supporting antenna diversity," in *Proc. IEEE ITA*, La Jolla, CA, USA, Jan. 2016, pp. 1–7.
- [5] J. Choi, V. Va, N. Gonzalez-Prelcic, R. Daniels, C. R. Bhat, and R. W. Heath, "Millimeter-wave vehicular communication to support massive automotive sensing," *IEEE Commun. Mag.*, vol. 54, no. 12, pp. 160–167, Dec. 2016.
- [6] Z. Qin, H. Ye, G. Y. Li, and B.-H. Juang, "Deep learning in physical layer communications," *IEEE Wireless Commun.*, vol. 26, no. 2, pp. 93–99, Apr. 2019.
- [7] Y. Wang, M. Narasimha, and R. W. Heath, "MmWave beam prediction with situational awareness: A machine learning approach," in *Proc. IEEE SPAWC*, Kalamata, Greece, June 2018, pp. 1–5.
- [8] C. Anton-Haro and X. Mestre, "Learning and data-driven beam selection for mmWave communications: An angle of arrival-based approach," *IEEE Access*, vol. 7, pp. 20 404–20 415, Jan. 2019.
- [9] X. Sun, C. Qi, and G. Y. Li, "Beam training and allocation for multiuser millimeter wave massive MIMO systems," *IEEE Trans. Wireless Commun.*, vol. 18, no. 2, pp. 1041–1053, Feb. 2019.
- [10] W. Ni, X. Dong, and W. S. Lu, "Near-optimal hybrid processing for massive MIMO systems via matrix decomposition," *IEEE Trans. Signal Process.*, vol. 65, no. 15, pp. 3922–3933, Aug. 2017.
- [11] H. Yan, V. Boljanovic, and D. Cabric, "Tracking sparse mmWave channel: Performance analysis under intra-cluster angular spread," in *Proc. IEEE SPAWC*, Kalamata, Greece, Dec. 2018, pp. 1–5.