Towards a Foundation Model for Communication Systems

Davide Buffelli*
MediaTek Research
London, UK
davide.buffelli@mtkresearch.com

Sowmen Das*
MediaTek Research
London, UK
sowmen.das@mtkresearch.com

Yu-Wei Lin*
MediaTek
Hsinchu, R.O.C.
yu-wei.lin@mediatek.com

Sattar Vakili*
MediaTek Research
Cambridge, UK
sattar.vakili@mtkresearch.com

Chien-Yi Wang
MediaTek
Hsinchu, R.O.C.
chien-yi.wang@mediatek.com

Masoud Attarifar
MediaTek
Cambridge, UK
masoud.attarifar@mediatek.com

Pritthijit Nath+
University of Cambridge
Cambridge, UK
pn341@cam.ac.uk

Da-shan Shiu
MediaTek Research
Taipei, R.O.C.
ds.shiu@mtkresearch.com

***Abstract*—Artificial Intelligence (AI) has demonstrated unprecedented performance across various domains, and its application to communication systems is an active area of research. While current methods focus on task-specific solutions, the broader trend in AI is shifting toward large *general* models capable of supporting multiple applications. In this work, we take a step toward a *foundation model for communication data*—a transformer-based, multi-modal model designed to operate directly on communication data. We propose methodologies to address key challenges, including tokenization, positional embedding, multimodality, variable feature sizes, and normalization. Furthermore, we empirically demonstrate that such a model can successfully estimate multiple features, including transmission rank, selected precoder, Doppler spread, and delay profile.**

***Index Terms*—AI, foundation models, transformer architecture, modem, communication channels, channel state information, Doppler spread, delay spread**

I. Introduction

In modern communication systems, numerous estimation and optimization tasks are required to ensure effective signal transmission and reception. Some examples include channel estimation, Doppler spectrum and delay spread estimation, and channel state information (CSI) feedback, which includes precoding matrix and its rank.

In the past few years, an increasingly large amount of research has focused on replacing traditional solutions to the above tasks with machine learning, and deep learning in particular [1], [2]. These methods are based on the idea of learning from data rather than relying on handcrafted heuristics, paving the way for more intelligent and efficient networks. Indeed, AI is already expected to become a key component of 6G networks [3] and will play an even more central role in future communication systems [4].

Current research in this domain has focused on designing task-specific machine learning solutions. These approaches involve collecting large amounts of supervised data, which is costly and time-consuming, and require a dedicated model for each task. In contrast, the broader field of AI is increasingly adopting a paradigm in which a single large model, referred to as a *foundation model*, is trained on vast amounts of unsupervised (and thus easier to obtain) data. Such a foundation model learns intricate relationships in the data, and is then leveraged to perform downstream tasks, by either applying it directly, specializing it with some additional fine-tuning, or by using its learned representations.

In this work, we move the first steps towards a *Foundation Model for Communication Systems*. This is a large deep learning model trained on vast quantities of unsupervised data coming from network communication. While foundation models have been proposed for domains like natural language, code, images, physical data, and time series, communication systems present unique challenges that are not addressed by the current literature. In more detail, dealing with multiple features at each slot in the input sequence, features from different domains (categorical, integer, real and complex valued, matrices), and with varying sizes and magnitudes, requires carefully designed ad-hoc procedures and components. We summarize our contributions as follows:

- We design and train a foundation model for communication systems addressing key challenges including tokenization, positional embedding, multimodality, features of varying size, and normalization.
- We design a simulation system to generate unsupervised data for the training of our foundation model.
- We test our foundation model in several scenarios, showing it can successfully perform multiple estimations.
- We report experimental results on how model size and dataset size affect estimation performance, providing initial insights into the scaling behavior of foundation models for communication systems (see Figures 2 and 3).
- We will release both the data generation pipeline and the model implementation, including training code, to further support research in this area.

*Made leading contributions to different aspects of the work. +Pritthijit Nath's work was part of his internship at MediaTek Research, UK.

II. Related Work

We focus our discussion on foundation models, and large language models (LLMs) applied to communication applications. The literature on machine learning for communication systems is rapidly growing, and a complete presentation would be out of the scope of this paper. We refer the interested reader to recent surveys on machine learning [5]–[7], and deep learning in particular [8], [9], in this domain.

A. Foundation Models

The term *foundation model* was introduced in the context of LLMs [10], i.e., models trained on large quantities of textual data. In recent years, we have observed a proliferation of LLMs (e.g. [11]–[15]), with increasingly capable performance (see recent surveys [16], [17] for more details).

Multimodal foundation models that combine data from different modalities have also been introduced. The most popular such models combine text with images [18]–[22], with few examples of other combinations like text and audio [23], [24] and text, audio, images, depth, thermal, and IMU data [25].

Recently, significant effort has been dedicated to creating foundation models outside of the traditional textual, visual, or audio domains. Examples of this are foundation models for signals from wearable devices [26], [27], foundation model for earth-related signals [28] and physical signals [29], foundation models for electrical power grid [30], and foundation models for time-series [31]–[37]. While these models share commonalities to our work, the setting of communication systems presents unique challenges that are not addressed by current literature. In particular, these works deal with limited amounts of features, and with low heterogeneity, which is the opposite of communication systems as we discuss in Section IV.

B. LLMs for Communication Systems

With the rise in popularity of LLMs, several works have focused on introducing them into the context of communication systems. [38] presents the challenges that need to be addressed to make existing pre-trained LLMs able to understand data from communication system. [39] introduce a dataset containing numerous technical documents that can be used to introduce knowledge related to communication systems to LLMs. Finally, [40] shows that "small" language models enhanced with Retrieval-Augmented Generation (RAG) [41] approaches can already understand communications data, and [42] further improve the RAG procedure for specialized technical documents. These works focus on improving the communications-related knowledge of LLMs; while these in turn can be used to develop solutions for specific problems, they do not directly operate on low-level data. In our work instead we aim at designing a model that directly operates on raw data and that can be used to directly solve specific downstream tasks.

III. Communication System Model and Simulator

In this section, we describe the communication system model and the simulated data used to train the foundation model.

A. Communication System Model

We consider a multiple-input multiple-output (MIMO) orthogonal frequency-division multiplexing (OFDM) system of N_T transmit (Tx) antennas, N_R receive (Rx) antennas, K subcarriers per OFDM symbol with subcarrier spacing f_{sc} Hz, and L OFDM symbols per slot. We restrict our attention to channel state information (CSI) acquisition, which is an important subsystem of wireless communications. For CSI acquisition, the transmitter sends pilot signals to the receiver. The receiver estimates the CSI and then reports to the transmitter.

The input–output in the frequency domain are related as

$$\mathbf{y}[k,l,n] = \mathbf{H}[k,l,n]\mathbf{x}[k,l,n] + \mathbf{z}[k,l,n],$$

where $\mathbf{y}[k,l,n] \in \mathbb{C}^{N_R}$ is the Rx signal vector at subcarrier k and symbol l in slot n, $\mathbf{x}[k,l,n] \in \mathbb{C}^{N_T}$ is the Tx signal vector, $\mathbf{H}[k,l,n] \in \mathbb{C}^{N_R \times N_T}$ is the channel matrix drawn from a wide-sense stationary random process of mean zero and unit power per entry, and $\mathbf{z}[k,l,n] \in \mathbb{C}^{N_R}$ is the additive noise independent of $\mathbf{H}$. We assume $\{\mathbf{z}[k,l,n]\}$ are independent and identically distributed (i.i.d.) circularly-symmetric complex Gaussian $\mathcal{CN}(\mathbf{0}, \mathbf{C}_{\text{n}})$, where $\mathbf{0}$ denotes an all-zero vector and $\mathbf{C}_{\text{n}}$ denotes the noise covariance matrix. For data transmission, each modulated symbol vector $\mathbf{s} \in \mathcal{A}^R$ is precoded by $\mathbf{W}_{(R)} \in \mathcal{W}_R \subset \mathbb{C}^{N_T \times R}$, where $\mathcal{A}$ is the set of constellation points, R denotes the rank (number of layers), and $\mathcal{W}_R$ is the codebook of rank R known by the transmitter and the receiver. We assume that $\mathbb{E}[\|\mathbf{s}\|^2] = P$ and $\|\mathbf{W}_{(R)}\|_F^2 = 1$ for all R, where $\|\cdot\|_F$ denotes the Frobenius norm. We assume that the same precoder is applied to all subcarriers and symbols within a slot. Then, we have $\mathbf{x}[k,l,n] = \mathbf{W}_{(R)}[n]\mathbf{s}[k,l,n]$ and thus the input–output relation can be expanded as

$$\mathbf{y}[k,l,n] = \mathbf{H}[k,l,n]\mathbf{W}_{(R)}[n]\mathbf{s}[k,l,n] + \mathbf{z}[k,l,n].$$

We divide the K subcarriers into B subcarrier groups of size M, so that $K = BM$. Denote $\mathcal{P} = \{(Mm, l) : m \in \{1, \cdots, B\}, l \in \mathcal{S}\}$, where $\mathcal{S}$ denotes the set of symbols with pilots. The performance of CSI acquisition can be assessed by spectral efficiency defined as

$$G(R, \mathbf{W}_{(R)}) = \frac{1}{|\mathcal{P}|} \sum_{(k,l)\in\mathcal{P}} \log\det\left(\mathbf{C}_{\text{n}} + P\mathbf{H}[k,l]\mathbf{W}_{(R)}\mathbf{W}_{(R)}^H\mathbf{H}^H[k,l]\right),$$

which follows the capacity of MIMO channel with CSI at receiver (see e.g., Section 8.2.1 of [43]).

To measure the channel matrices $\{\mathbf{H}[k,l]\}$, we transmit pilots following a comb-type structure: For each Tx antenna j, we set $\mathbf{x}[Mm + j, l, n] = \sqrt{P}\mathbf{e}_j$, where $l \in \mathcal{S}$ and $\mathbf{e}_j$ denotes the standard unit vector with one on the j-th position and zero otherwise. Besides, the noise covariance $\mathbf{C}_{\text{n}}$ needs

to be estimated, so we set $\mathbf{x}[Mm+j,l,n]=\mathbf{0}$ for $j=N_T+1,\cdots,N_T+N_Z$ and assume $N_Z=M-N_T$.

The receiver needs to determine $(R,\mathbf{W}_{(R)})$ from the received signals at pilot subcarriers, also known as channel frequency response (CFR): For $j\in\{1,\cdots,N_T\}$,

$$\begin{aligned}\tilde{\mathbf{h}}_j[m,l,n]&\triangleq\mathbf{y}[Mm+j,l,n]\\&=\sqrt{P}\mathbf{h}_j[Mm+j,l,n]+\mathbf{z}[Mm+j,l,n],\end{aligned}$$

where $\mathbf{h}_j$ is the column j of $\mathbf{H}$, and for $k\in\{1,\cdots,BN_Z\}$,

$$\tilde{\mathbf{z}}[k,l,n]\triangleq\mathbf{z}[M\lfloor(k-1)/N_Z\rfloor+((k-1)\bmod N_Z)+1,l,n].$$

To simplify notation, we assume per-slot operation and hence omit the slot index n in the remainder of this section.

1) Noise covariance estimation: The noise covariance can be estimated by

$$\hat{\mathbf{C}}_{\text{n}}=\frac{1}{BN_Z|\mathcal{S}|}\sum_{k=1}^{BN_Z}\sum_{l\in\mathcal{S}}\tilde{\mathbf{z}}[k,l]\tilde{\mathbf{z}}^H[k,l].$$

The average noise power at Rx antenna i is denoted as $\hat{\sigma}_i^2\triangleq[\hat{\mathbf{C}}_{\text{n}}]_{ii}$.

2) Signal power estimation: The signal power P can be estimated as

$$\hat{P}=\frac{1}{N_TB|\mathcal{S}|N_R}\sum_{j=1}^{N_T}\sum_{m=1}^{B}\sum_{l\in\mathcal{S}}\left\|\tilde{\mathbf{h}}_j[m,l]\right\|^2-\frac{1}{N_R}\sum_{i=1}^{N_R}\hat{\sigma}_i^2.$$

It is desirable to perform channel denoising before determining the rank and precoder. We perform robust channel estimation by assuming the power delay profile and the Doppler spectrum are both of rectangular shape [44].

3) Delay profile estimation: For robust channel estimation in frequency domain, we assume that any two channel gains separated by Δ pilots in the same OFDM symbol, say $h_{(f),m}$ and $h_{(f),m+\Delta}$, satisfy that

$$\begin{aligned}&\frac{\mathbb{E}[h_{(f),m+\Delta}h^*_{(f),m}]}{\sqrt{\mathbb{E}[|h_{(f),m+\Delta}|^2]\mathbb{E}[|h_{(f),m}|^2]}}\\&=e^{-j2\pi\mu\Delta Mf_{\text{sc}}}\frac{\sin(\pi\ell\Delta Mf_{\text{sc}})}{\pi\ell\Delta Mf_{\text{sc}}},\end{aligned}$$

where μ and ℓ are the center and the length of the delay profile, respectively. The center and the length are estimated by the following procedure: For the tuple of Tx antenna j, Rx antenna i, and symbol l, we collect its associated CFR into a vector $\tilde{\mathbf{h}}_{(f)}[i,j,l]$ and transform the CFR into the channel impulse response (CIR) $\tilde{\mathbf{h}}_{(t)}[i,j,l]$ through inverse discrete Fourier transform (IDFT) after zero padding to size N_{FFT}, which is the smallest power of 2 larger than K. The noisy delay profile $\tilde{\mathbf{p}}_{(t)}[i]$ at Rx antenna i is then calculated as

$$\tilde{\mathbf{p}}_{(t)}[i]=\frac{1}{N_T|\mathcal{S}|}\sum_{j=1}^{N_T}\sum_{l\in\mathcal{S}}\left|\tilde{\mathbf{h}}_{(t)}[i,j,l]\right|^2.$$

Only the tap(s) with power larger than $3\hat{\sigma}_i^2$ are considered as including the desired signal. Denote $\mathcal{D}[i]=\left\{n\middle|\left[\tilde{\mathbf{p}}_{(t)}[i]\right]_n>3\hat{\sigma}_i^2,0\le n<N_{\text{FFT}}\right\}$. Then, the starting and ending positions of the delay profile $(\hat{n}_{\text{start}}[i],\hat{n}_{\text{end}}[i])$ are found as the pair that covers $\mathcal{D}[i]$ with the minimum length, considering invariance under circular shift. Finally, we have

$$\begin{aligned}\hat{\mu}[i]&=\frac{1}{Mf_{\text{sc}}N_{\text{FFT}}}\left(\frac{\hat{n}_{\text{start}}[i]+\hat{n}_{\text{end}}[i]}{2}\right),\\\hat{\ell}[i]&=\frac{1}{Mf_{\text{sc}}N_{\text{FFT}}}(\hat{n}_{\text{start}}[i]-\hat{n}_{\text{end}}[i]+1),\end{aligned}$$

and the estimated correlation matrix $\hat{\mathbf{R}}_{\text{f,robust}}$ with

$$\begin{aligned}&[\hat{\mathbf{R}}_{\text{f,robust}}]_{m_1,m_2}\\&=e^{-j2\pi\hat{\mu}[i](m_1-m_2)Mf_{\text{sc}}}\frac{\sin(\pi\hat{\ell}[i](m_1-m_2)Mf_{\text{sc}})}{\pi\hat{\ell}[i](m_1-m_2)Mf_{\text{sc}}}.\end{aligned}$$

In case that $\mathcal{D}[i]$ is empty, we set $\hat{\mu}[i]=\frac{\arg\max_n[\tilde{\mathbf{p}}_{(t)}[i]]_n}{Mf_{\text{sc}}N_{\text{FFT}}}$ and $\hat{\ell}[i]=\frac{1}{Mf_{\text{sc}}N_{\text{FFT}}}$. Following the above procedure, for each Rx antenna i, the genie center $\mu[i]$ and the genie length $\ell[i]$ for the noiseless delay profile $\mathbf{p}_{(t)}[i]$ can be calculated. With an abuse of notation, we denote by $\mathbf{R}_{\text{f,robust}}[i]$ the genie robust frequency correlation matrix.

4) Doppler spectrum estimation: For robust channel estimation in time domain, we assume that any two channel gains $h_{(t),l_1}$ and $h_{(t),l_2}$ of the same subcarrier in symbol l_1 and l_2, respectively, satisfy that

$$\frac{\mathbb{E}[h_{(t),l_1}h^*_{(t),l_2}]}{\sqrt{\mathbb{E}[|h_{(t),l_1}|^2]\mathbb{E}[|h_{(t),l_2}|^2]}}=\frac{\sin(\pi w(l_1-l_2)T}{\pi w(l_1-l_2)T},$$

where w is the width of the Doppler spectrum and T is the OFDM symbol duration. For the tuple of Tx antenna j, Rx antenna i, and subcarrier group m, we collect its associated received signals into a vector $\tilde{\mathbf{h}}_{(t)}[i,j,m]$. Then, the time covariance matrix of Rx antenna i can be estimated as

$$\hat{\mathbf{C}}_{\text{time}}[i]=\frac{1}{N_TB}\sum_{j=1}^{N_T}\sum_{m=1}^{B}\tilde{\mathbf{h}}_{(t)}[i,j,m]\tilde{\mathbf{h}}^H_{(t)}[i,j,m]-\hat{\sigma}_i^2\mathbf{I}_{|\mathcal{S}|}.$$

The corresponding estimated time correlation matrix is denoted by $\hat{\mathbf{R}}_{\text{time}}[i]$, which can be calculated from the entries of $\hat{\mathbf{C}}_{\text{time}}$ by $r_{ij}=c_{ij}/\sqrt{c_{ii}c_{jj}}$. Then, the width w can be found by

$$\hat{w}[i]=\arg\min_w\|\mathbf{R}_{\text{t,robust}}(w)-\hat{\mathbf{R}}_{\text{time}}[i]\|_F^2,$$

where

$$[\mathbf{R}_{\text{t,robust}}(w)]_{l_1,l_2}=\frac{\sin(\pi w(l_1-l_2)T}{\pi w(l_1-l_2)T}.$$

Denote $\hat{\mathbf{R}}_{\text{t,robust}}[i]=\mathbf{R}_{\text{t,robust}}(\hat{w}[i])$ the estimated robust time correlation matrix of Rx antenna i. The genie time covariance matrix $\mathbf{C}_{\text{time}}[i]$ is calculated by replacing $\tilde{\mathbf{h}}_{(t)}[i,j,m]$ and $\hat{\sigma}_i^2$ by $\mathbf{h}_{(t)}[i,j,m]$ and 0, respectively. Following the above procedure, the genie width $w[i]$ and the genie time correlation matrix $\mathbf{R}_{\text{time}}[i]$ can be calculated. We denote by $\mathbf{R}_{\text{t,robust}}[i]=\mathbf{R}_{\text{t,robust}}(w[i])$ the genie robust time correlation matrix.

Once we have $\hat{\mathbf{R}}_{\text{f,robust}}[i]$ and $\hat{\mathbf{R}}_{\text{t,robust}}[i]$, the robust channel estimation is performed as follows: For Rx antenna i and Tx antenna j,

$$\left[\hat{\mathbf{h}}_f[i,j,l]\right]_{l\in\mathcal{S}} = \hat{\mathbf{R}}_{\text{robust}}[i]\left(\frac{\hat{\sigma}_i^2}{\hat{P}}\mathbf{I}+\hat{\mathbf{R}}_{\text{robust}}[i]\right)^{-1}\left[\tilde{\mathbf{h}}_f[i,j,l]\right]_{l\in\mathcal{S}},$$

where $\mathbf{R}_{\text{robust}}[i] = \mathbf{R}_{\text{t,robust}}[i] \otimes \mathbf{R}_{\text{f,robust}}[i]$, $\otimes$ is the Kronecker product, and $[\tilde{\mathbf{h}}_f(i,j,l)]_{l\in\mathcal{S}}$ denotes the concatenation of CFRs $\tilde{\mathbf{h}}_f[i,j,l]$ for all l in $\mathcal{S}$.

5) Precoder selection: We denote by $\hat{\mathbf{H}}[m,l]$ the estimated channel matrix of subcarrier group m and symbol l. The average whitened spatial covariance matrix is given by

$$\hat{\mathcal{C}}_S = \frac{1}{B|\mathcal{S}|}\sum_{m=1}^{B}\sum_{l\in\mathcal{S}}\hat{\mathbf{H}}^H[m,l]\hat{\mathbf{C}}_{\text{n}}^{-1}\hat{\mathbf{H}}[m,l].$$

For each rank R, the precoder $\mathbf{W}_{(R)}$ is selected as

$$\hat{\mathbf{W}}_{(R)} = \underset{\mathbf{W}\in\mathcal{W}_R}{\arg\max}\log\det\left(\mathbf{I}_R+\mathbf{W}^H\hat{\mathbf{C}}_S\mathbf{W}\right).$$

6) Rank selection: The reported rank $\hat{R}$ is selected as

$$\hat{R} = \underset{R\in\{1,\cdots,N_R\}}{\arg\max}\log\det\left(\mathbf{I}_R+\hat{\mathbf{W}}_{(R)}^H\hat{\mathbf{C}}_S\hat{\mathbf{W}}_{(R)}\right).$$

B. Dataset

We use the open-source SIONNA package [45], together with the system model described in this section, to generate the dataset. At each time step n, corresponding to a slot, we record a selected set of diverse features to conduct proof-of-concept experiments. The features are listed in Table I.

TABLE I
SELECTED FEATURES

Feature	Description
Channel type (categorical)	The type of SIONNA channel (UMi, UMa, RMa)
$K \in \mathbb{Z}^+$	Number of subcarriers per OFDM symbol
$\hat{\mathbf{C}}_{\text{n}} \in \mathbb{C}^{N_R\times N_R}$	Estimated noise covariance matrix
$\hat{\mathbf{R}}_{\text{f, robust}} \in \mathbb{C}^{B\times B}$	Estimated frequency correlation matrix
$\hat{\mathbf{C}}_{\text{time}} \in \mathbb{C}^{\lvert\mathcal{S}\rvert\times\lvert\mathcal{S}\rvert}$	Estimated time covariance matrix
$\hat{\mathbf{R}}_{\text{time}} \in \mathbb{C}^{\lvert\mathcal{S}\rvert\times\lvert\mathcal{S}\rvert}$	Estimated time correlation matrix
$\hat{\mu}, \hat{\ell} \in \mathbb{R}$	Estimated center and length of delay profile
$\hat{w} \in \mathbb{R}^+$	Estimated width of Doppler spectrum
$\hat{\mathbf{W}}_{(R)} \in \mathbb{C}^{N_T\times R}$	Selected precoder of rank R
$\hat{R} \in \mathbb{Z}^+$	Selected transmission rank
$\hat{G} \in \mathbb{R}^+$	Estimated Spectral Efficiency

The symbol duration and cyclic prefix length follow the fifth generation (5G) New Radio (NR) numerology [46]. Specifically, we use a normal cyclic prefix. In NR, a slot consists of 14 OFDM symbols, hence $L = 14$. The configurations used to generate the dataset is listed in Table II, wherein SNR is randomly drawn following uniform distribution. To reduce implementation complexity, the parameters related to $(\hat{w}, \hat{\mu}, \hat{l})$ are selected from predefined candidate sets.

Considering all possible combinations for the values in Table II, there is a total of 86403 unique simulation settings.

TABLE II
DATASET CONFIGURATION

Parameter	Values
Channel type	UMi, UMa, RMa
(Center frequency, subcarrier spacing) (Hz)	(2.6G, 15k), (3.5G, 30k)
SNR (dB range)	[0, 30]
Receiver speed (km/h)	3, 10, 30, 60, 90
(N_T, N_R)	(4, 1), (4, 2), (4, 4), (8, 1), (8, 2), (8, 4)
Number of subcarrier group (B)	25, 50, 75, 100
Size of subcarrier group (M)	12, 24, 48
Set of pilot symbols ($\mathcal{S}$)	{2, 8}, {2, 6, 10}, {4, 8, 12}, {2, 5, 8, 11}

As generating data for all configurations would be intractable, we randomly select a subset of 29000 configurations. For each selected configuration we run the simulation 8 times, each time with a different SNR (sampled randomly with uniform distribution from the range specified in the above table) for 100 slots. Each simulation uses a unique random seed. We then divide the obtained data into 5-slot sequences, each one representing one datapoint to be used as input for our model.

IV. METHODOLOGY

The setting of communication systems presents several challenges that hinder the development of a foundation model. These systems are inherently complex and involve a large number of interdependent variables. Factors such as the choice of modulation, channel conditions (e.g., signal-to-noise ratio, multipath propagation, Doppler shift, fading), and the internal states of components like transmitters and receivers all influence system behavior. Moreover, the data in communication systems is highly heterogeneous. Some variables vary across consecutive transmissions over the same channel, while others remain constant for a given channel but change across different channels. Variables can be categorical, scalar, vector, or matrix-valued; their dimensionality may vary with system configuration, and they may belong to either the real or complex domain.

These characteristics require a foundation model for communication systems to reason over and capture relationships among a large number of diverse and heterogeneous variables. In the remainder of this section, we present our methodology for training such a model, covering data pre-processing, model design, and pre-training strategy.

a) Pre-processing: Two important aspects must be considered. The first is that for multi-dimensional features, the model must handle varying dimensionality depending on the current system settings. Since neural networks require fixed input sizes, we identify the maximum dimensionality for each feature and apply zero-padding to those with smaller dimensions. Padding masks are then introduced in the neural network computations to ensure that the model ignores the padded values.

The second aspect is that different features have different magnitudes, which can further vary significantly depending on the system configuration. We adopt a normalization strategy tailored to the properties of each feature. Scalar features are

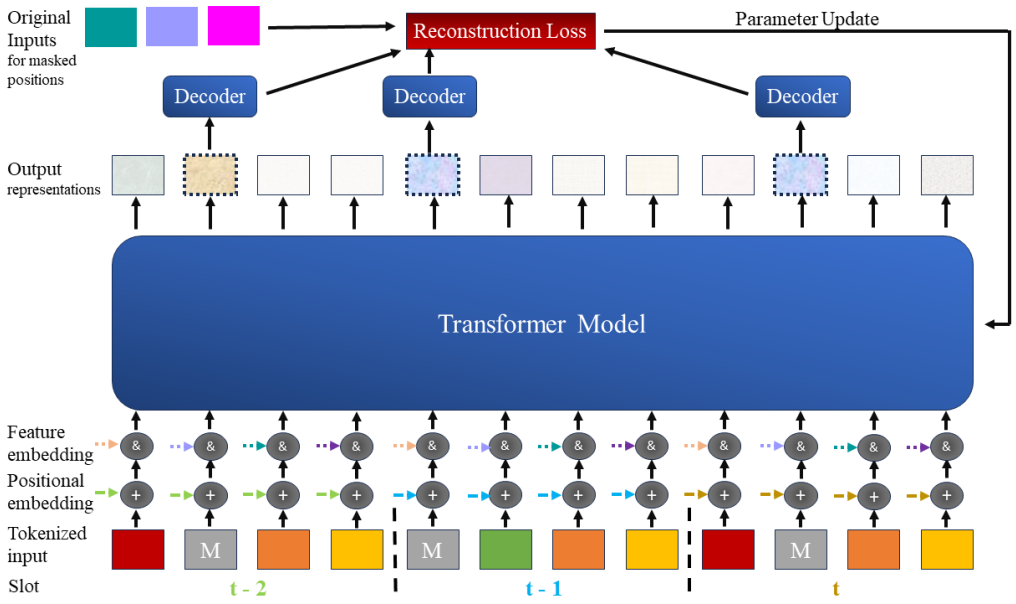

Fig. 1. **Scheme of our foundation model for communications data and the pre-training procedure.** Tokens for each feature are first added to a positional embedding; features at the same input slot receive the same positional embedding (indicated by the color of the arrows). A feature embedding is then concatenated (&); the tokens for the same feature at different slots receive the same feature embedding. The transformer processes these tokens and outputs a representation for each input. During pre-training, a subset of input features is masked (denoted as "M"). The output representations corresponding to the masked inputs are decoded into the original feature space, and a reconstruction loss is computed. The model is trained to minimize this loss.

normalized using global statistics—specifically, we compute the mean and standard deviation over the training set and use them for normalization. Matrix and vector features are handled individually: we either normalize them by computing statistics across the input sequence or leave them unchanged (e.g., correlation matrices, which are already normalized). For the features that require padding, the normalization statistics are computed on the unpadded elements (and the normalization is applied only on those elements).

b) Tokenization: Given the heterogeneity of the features in our data, we divide them into four categories: scalars, vectors, matrices, and categorical. We design a tokenization method for each category as follows:

- **Scalars:** we use Fourier encodings: $\mathrm{Tok}(x) = \left[\cos\frac{2\pi x}{\lambda_i}, \sin\frac{2\pi x}{\lambda_i}\right]$ for $0 \le i < D/2$, where the λ_i are logarithmically spaced values taken from an interval $[\lambda_{\min}, \lambda_{\max}]$, chosen based on the scale and variability of the feature.
- **Vectors:** given a vector $x \in \mathbb{R}^d$, we apply a learnable linear transformation Wx, where $W \in \mathbb{R}^{D\times d}$.
- **Matrices:** following the approach from ViT [47], we divide a matrix $X \in \mathbb{R}^{d_1\times d_2}$ into "patches": $\{X_j^{(p)} \in \mathbb{R}^{p_1\times p_2}, j = 1,\dots,z\}$, where p_1 and p_2 are patch sizes, and $z = (d_1/p_1)\cdot(d_2/p_2)$ is the number of patches. Each patch is then flattened into a vector of size $p = p_1\cdot p_2$ and transformed using a linear map $W \in \mathbb{R}^{D\times p}$. A matrix is thus converted into z tokens, providing finer granularity to encode its structure. We define patch sizes using a simple heuristic: we choose the size such that the number of tokens per matrix is at most 64, with a minimum patch size of 8×8. Alternatively, the patch size could be treated as a tunable hyperparameter, but we avoid this extra cost as our heuristic performs well in our experiments.
- **Categorical:** we assign a learnable vector $V \in \mathbb{R}^D$ to each category.

To handle complex-valued features, we first normalize and pad them as needed, and then convert them into two-channel vectors or matrices by stacking the real and imaginary parts.

c) Transformer Model: For the base architecture of the transformer, we follow the Llama model [48], as adopted by several state-of-the-art open-source models. This variant of the transformer architecture uses RoPE [49] for positional embeddings, RMS Normalization [50] as normalization in between layers, and SwiGLU [51] non-linearity in the feed-forward components of the Transformer blocks.

Typically, transformer-based foundation models receive a single quantity (and thus a single token) per each element of the input sequence. In our case, however, we have multiple features—and hence multiple tokens—at each slot in the input sequence. We assign the same positional embedding to the tokens for all features within the same slot. This setup, however, means that the model cannot, by default, distinguish tokens corresponding to the same feature across different slots, as it has no information about which token, within those for a given slot, is related to which feature. To address this, we introduce a *feature embedding*: a learnable vector associated with each input feature, which is concatenated to the corresponding token to let the model recognize instances of the same feature across slots.

Once the tokens have been processed by the transformer, we

obtain an embedding vector for each input token. To compute the loss for a given feature, we need to decode the embedding vector(s) back into the original space of that feature. We design the decoding mechanism to follow an "inverse" process to the one used for tokenization:

- **Scalars**: a learnable vector $U \in \mathbb{R}^{1\times D}$ maps the hidden representation to a scalar
- **Vectors**: a learnable matrix $Q \in \mathbb{R}^{d\times D}$ maps the hidden representation to the original dimensionality
- **Matrices**: the token for each patch is mapped back to its original dimensionality with a learnable linear transformation $E \in \mathbb{R}^{p_1\cdot p_2\times D}$. The patches are then reshaped into a matrix in $\mathbb{R}^{p_1\times p_2}$ and concatenated back into the original shape for the feature
- **Categorical**: a learnable matrix $R \in \mathbb{R}^{n_c\times D}$ maps the hidden representation to a 1-hot encoding of the category

The learnable components of the decoding mechanism are intentionally kept simple to encourage the model to capture complexity within the representations themselves, rather than in the decoding functions.

d) Pre-Training Procedure: Our goal is to enable the foundation model to learn relationships between features. To this end, we adopt a self-supervised approach (requiring no annotations) based on masked token prediction, similar to the pretraining strategy used in language models such as BERT [11]. While training using *random masking* is common in language models, communication data requires more care. At each position in the input sequence, multiple features are present, and some features are not predictable from others. Naively masking such features can introduce instability during training.

To address this, we identify a subset of five *target* features from Table I—namely, the transmission rank, selected precoder, Doppler spectrum, center and length of the delay profile. These are features that can be estimated from the remaining ones, as discussed in Section III. Pretraining is then performed by randomly masking a subset of the target features at each slot in the input sequence and training the model to predict them. An overview of the model and pretraining procedure is provided in Figure 1.

V. Experiments

The evaluation of our foundation model focuses on demonstrating its effectiveness in understanding communication systems data and its ability to perform estimation tasks. Specifically, we consider two scenarios:

1. **Forecasting.** The model is tasked with estimating the value of a given feature in the next slot.
2. **Interpolation.** We randomly mask the values of a feature at certain slots in the input sequence, and the model is tasked with estimating the values of the masked feature. In more detail, we perform one evaluation run for each feature. During each run, for each batch, a slot of the input sequence is selected randomly (with uniform distribution), and the selected feature is masked for that slot.

A. Experimental Setting

Data. We generate 1 million datapoints using the simulator described in Section III-B. The data is then randomly split into training, validation, and test sets, containing 80%, 10%, and 10% of the data, respectively.

Foundation Model Evaluation Procedure. As described above, we evaluate the model by measuring its ability to estimate missing values for features of interest through the tasks of forecasting and interpolation. The features used in these tasks are the target features defined in Section IV: transmission rank, selected precoder, Doppler spread, center and length of the delay profile.

Hyperparameters. We use the validation set to tune the hyperparameters of our model. Specifically, we perform a grid search over learning rate, learning rate scheduling, token dimension, number of layers, and number of attention heads. The values explored for the tuning procedure, and the final values used for our experiments are presented in Appendix A. The final model we use for the estimation results has 75 layers, 24 attention heads, a token dimension of 387, and a total of 100 million parameters,

Computational Resources. For data generation we found that it takes 0.5 seconds to generate a single slot. We parallelize the data generation over multiple Intel Xeon Platinum processing nodes. For training our largest model, we use 2× NVIDIA A6000 GPUs.

B. Results

Table III presents the results for the *forecasting* and *interpolation* tasks. The model achieves consistently low mean squared error (MSE) across all five target features, demonstrating its ability to capture dependencies in the communication data. Since all features are normalized prior to training, the reported MSE values are computed on standardized scales and can be interpreted as percentage errors. As expected, interpolation performs slightly better than forecasting, as it estimates missing features using other available features within the same slot, while forecasting requires predicting feature values at future slots.

TABLE III
Estimation Error (MSE) for the **forecasting** and **interpolation** tasks on five key features: center of the delay profile ($\hat{\mu}$), length of the delay profile ($\hat{\ell}$), Doppler spectrum width ($\hat{w}$), transmission rank ($\hat{R}$), and selected precoder ($\hat{\mathbf{W}}$).

	$\hat{\mu}$	$\hat{\ell}$	$\hat{w}$	$\hat{R}$	$\hat{\mathbf{W}}$
Forecasting	0.019	0.021	0.077	0.129	0.101
Interpolation	0.019	0.020	0.054	0.128	0.100

Among the features, Doppler spectrum width ($\hat{w}$) and delay profile parameters ($\hat{\mu}$, $\hat{\ell}$), which are all scalar-valued, are estimated with the highest accuracy. In contrast, transmission rank ($\hat{R}$) and especially the selected precoder ($\hat{\mathbf{W}}$)—a matrix-valued feature with variable dimensionality—exhibit higher errors. This reflects the increased difficulty of modeling high-dimensional, structured, and variable-size features. These

observations are consistent with the challenges discussed in Section IV and highlight the model's ability to generalize across diverse modalities in communication systems data.

C. Scaling Behaviour.

We carried out experiments to study how model performance scales with compute, measured in floating point operations (FLOPs) allocated to data generation and training (we show how to compute FLOPs in Appendix A). Specifically, we trained foundation models of three sizes (5M, 30M, and 100M parameters) on datasets of varying sizes (10^6, $5 \cdot 10^6$, and 10^7 training examples). These datasets are obtained by randomly subsampling the training set used for the estimation experiments. Each configuration was trained for up to 20 epochs.

Figure 2 shows how the test loss (sum of the losses for the target features) varies with compute allocated to data generation, which determines dataset size. Figure 3 shows test loss as a function of training compute, which controls the number of model updates. Performance improves with both model and dataset size, especially when scaled together. These trends are consistent with known scaling behaviour observed in other domains. Larger models benefit more from extended training. The results highlight the importance of co-scaling model size, data, and training budget to fully realize performance gains.

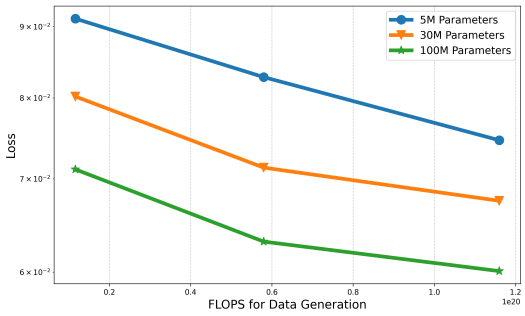

Fig. 2. Scaling behaviour vs. data generation compute. Each curve shows the test loss for a model of a given size (5M, 30M, or 100M parameters) as a function of compute used for data generation.

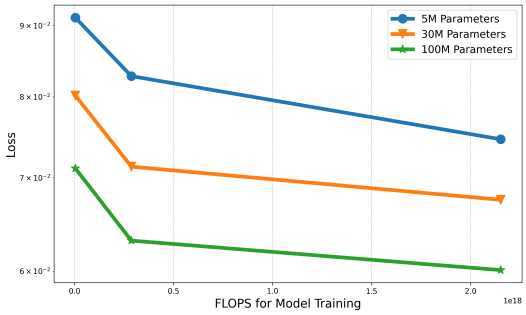

Fig. 3. Scaling behaviour vs. training compute. Each curve shows the test loss for models of a given size (5M, 30M, 100M) as a function of compute used for training.

VI. Conclusion

In this work, we laid the groundwork for a foundation model for communication systems. We discussed and addressed the unique challenges involved in designing such a model, developed a simulation system to generate training data, and demonstrated the model's ability to perform forecasting and interpolation across several key features.

Future work will focus on scaling up both the dataset and model size, and incorporating additional features. We will release our data generation pipeline and model upon acceptance of this paper, with the hope that this work serves as a first step in this paradigm shift. We aim to support the community's progress toward building datasets, models, and benchmarks that advance the development of effective foundation models for communication systems, with potential applications in 6G.

References

[1] I. Žeger and G. Šišul, "Introduction to deep learning possibilities in communication systems," in *International Symposium ELMAR*, pp. 21–24, 2021.

[2] K. Thakkar, A. Goyal, and B. Bhattacharyya, "Emergence of deep learning as a potential solution for detection, recovery and de-noising of signals in communication systems," *International Journal of Intelligent Networks*, vol. 1, pp. 119–127, 2020.

[3] Y. Wang *et al.*, "Transformer-empowered 6g intelligent networks: From massive mimo processing to semantic communication," *Wireless Communications*, vol. 30, p. 127–135, Dec. 2023.

[4] W. Saad *et al.*, "Artificial general intelligence (agi)-native wireless systems: A journey beyond 6g," 2024. *arxiv:2405.02336*.

[5] Y. Sun, H. Lee, and O. Simpson, "Machine learning in communication systems and networks," *Sensors*, vol. 24, no. 6, 2024.

[6] S. Wang and G. Y. Li, "Machine learning in communications: A road to intelligent transmission and processing," 2024. *arxiv:2407.11595*.

[7] M. Q. Hamdan *et al.*, "Recent advances in machine learning for network automation in the o-ran," *Sensors*, vol. 23, no. 21, 2023.

[8] W. Yu, F. Sohrabi, and T. Jiang, "Role of deep learning in wireless communications," *IEEE BITS the Information Theory Magazine*, vol. 2, no. 2, pp. 56–72, 2022.

[9] F. Liao, S. Wei, and S. Zou, "Deep learning methods in communication systems: A review," *Journal of Physics: Conference Series*, vol. 1617, p. 012024, aug 2020.

[10] R. Bommasani *et al.*, "On the opportunities and risks of foundation models," 2021. *arxiv:2108.07258*.

[11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4171–4186, Association for Computational Linguistics, 2019.

[12] T. Brown *et al.*, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems* (H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, eds.), vol. 33, pp. 1877–1901, Curran Associates, Inc., 2020.

[13] M. Lewis *et al.*, "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, eds.), (Online), pp. 7871–7880, Association for Computational Linguistics, July 2020.

[14] DeepSeek-AI *et al.*, "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning," 2025. *arxiv:2501.12948*.

[15] Y. Zhao *et al.*, "Marco-o1: Towards open reasoning models for open-ended solutions," 2024. *arxiv:2411.14405*.

[16] S. Minaee *et al.*, "Large language models: A survey," 2024. *arxiv:2402.06196*.

[17] W. X. Zhao *et al.*, "A survey of large language models," 2024. *arxiv:2303.18223*.

[18] J.-B. Alayrac *et al.*, "Flamingo: a visual language model for few-shot learning," in *Advances in Neural Information Processing Systems*, 2022.
[19] Z. Chen *et al.*, "Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling," 2025. *arxiv:2412.05271*.
[20] W. Dai *et al.*, "Nvlm: Open frontier-class multimodal LLMs," 2024. *arxiv:2409.11402*.
[21] G. Team *et al.*, "Gemini: A family of highly capable multimodal models," 2024. *arxiv:2312.11805*.
[22] OpenAI, "Openai o1 system card," 2024. https://openai.com/index/openai-o1-system-card/.
[23] H. Liu *et al.*, "Audioldm 2: Learning holistic audio generation with self-supervised pretraining," *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, vol. 32, p. 2871–2883, May 2024.
[24] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," 2022. *arxiv:2212.04356*.
[25] R. Girdhar, A. El-Nouby, Z. Liu, M. Singh, K. V. Alwala, A. Joulin, and I. Misra, "Imagebind: One embedding space to bind them all," in *CVPR*, 2023.
[26] G. Narayanswamy *et al.*, "Scaling wearable foundation models," in *International Conference on Learning Representations*, 2025.
[27] S. Abbaspourazad *et al.*, "Large-scale training of foundation models for wearable biosignals," in *International Conference on Learning Representations*, 2024.
[28] C. Bodnar *et al.*, "A foundation model for the earth system," 2024. *arxiv:2405.13063*.
[29] J. Lien *et al.*, "A phenomenological ai foundation model for physical signals," 2024. *arxiv:2410.14724*.
[30] H. F. Hamann *et al.*, "Foundation models for the electric power grid," 2024. *arxiv:2407.09434*.
[31] M. Goswami, K. Szafer, A. Choudhry, Y. Cai, S. Li, and A. Dubrawski, "Moment: A family of open time-series foundation models," in *International Conference on Machine Learning*, 2024.
[32] K. Rasul *et al.*, "Lag-Llama: Towards foundation models for probabilistic time series forecasting," 2024. *arxiv:2310.08278*.
[33] X. Shi, S. Wang, Y. Nie, D. Li, Z. Ye, Q. Wen, and M. Jin, "Time-moe: Billion-scale time series foundation models with mixture of experts," in *International Conference on Learning Representations*, 2025.
[34] A. Das, W. Kong, R. Sen, and Y. Zhou, "A decoder-only foundation model for time-series forecasting," 2024. *arxiv:2310.10688*.
[35] X. Liu *et al.*, "Moirai-moe: Empowering time series foundation models with sparse mixture of experts," 2024. *arxiv:2410.10469*.
[36] V. Ekambaram *et al.*, "Tiny time mixers (TTMs): Fast pre-trained models for enhanced zero/few-shot forecasting of multivariate time series," in *Conference on Neural Information Processing Systems*, 2024.
[37] Y. Liu *et al.*, "Sundial: A family of highly capable time series foundation models," 2025. *arxiv:2502.00816*.
[38] J. Shao *et al.*, "Wirelessllm: Empowering large language models towards wireless intelligence," *Journal of Communications and Information Networks*, vol. 9, no. 2, pp. 99–112, 2024.
[39] R. Nikbakht, M. Benzaghta, and G. Geraci, "Tspec-llm: An open-source dataset for llm understanding of 3gpp specifications," 2024. *arxiv:2406.1768*.
[40] N. Piovesan, A. D. Domenico, and F. Ayed, "Telecom language models: Must they be large?," 2024. *arxiv:2403.04666*.
[41] P. Lewis *et al.*, "Retrieval-augmented generation for knowledge-intensive nlp tasks," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, (Red Hook, NY, USA), Curran Associates Inc., 2020.
[42] A.-L. Bornea, F. Ayed, A. D. Domenico, N. Piovesan, and A. Maatouk, "Telco-rag: Navigating the challenges of retrieval-augmented language models for telecommunications," 2024. *arxiv:2404.15939*.
[43] D. Tse and P. Viswanath, *Fundamentals of wireless communication*. USA: Cambridge University Press, 2005.
[44] Y. Li, L. Cimini, and N. Sollenberger, "Robust channel estimation for ofdm systems with rapid dispersive fading channels," *IEEE Transactions on Communications*, vol. 46, no. 7, pp. 902–915, 1998.
[45] J. Hoydis *et al.*, "Sionna: An open-source library for next-generation physical layer research," 2022. *arxiv:2203.118548*.
[46] *NR; Physical Channels and Modulation, 3GPP Standard TS 38.211*, 2018.
[47] A. Dosovitskiyet *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2021.
[48] A. Grattafiori *et al.*, "The Llama 3 herd of models," 2024. *arxiv:2407.21783*.
[49] J. Su, M. Ahmed, Y. Lu, S. Pan, W. Bo, and Y. Liu, "Roformer: Enhanced transformer with rotary position embedding," *Neurocomput.*, vol. 568, Feb. 2024.
[50] B. Zhang and R. Sennrich, *Root mean square layer normalization*. Red Hook, NY, USA: Curran Associates Inc., 2019.
[51] N. Shazeer, "Glu variants improve transformer," 2020. *arxiv:2002.05202*.

APPENDIX

HYPERPARAMETER INFORMATION

Tuning. We provide the values used for the hyperparameter tuning procedure in Table IV.

Final Setting. The final hyperparameter values used for our experiments are shown in Table V. During pre-training, we randomly (with uniform distribution) mask one target feature at each input slot. We use a batch size of 256, a learning rate of 10^{-6}, and no learning rate schedule for all model and dataset sizes.

FLOPS CALCULATION

In this section we present how to compute the number of floating point operations (FLOPs) for model training and data generation.

A. Model Training

For the purpose of our calculation, the transformer is composed of an embedding layer, and then several transformer layers. Each transformer layer is composed of a self-attention mechanism, and a feed-forward network (other operations like normalization can be ignored due their much lower computational impact).

We perform our computation on the forward pass, as the ones for the backwards are roughly the same (i.e., the final

TABLE IV
VALUES FOR HYPERPARAMETER TUNING PROCEDURE.

Feature	Values
Learning Rate	$10^{-4}, 10^{-5}, 10^{-6}$
Learning Rate Schedule	None, Step, Cosine Annealing
Token Dimension	128, 256, 512
Number of Layers	8, 12, 16
Number of Attention Heads	4, 8, 16, 32

TABLE V
PARAMETERS AND DIMENSIONS FOR DIFFERENT MODEL SIZES USED IN OUR EXPERIMENTS.

Model	#layers	#head	token dim	hidden dim
5M	6	6	387	1548
30M	32	18	387	1548
100M	75	24	429	1716

TABLE VI
FLOPS FOR DATA GENERATION.

Module	FLOPs
Noise covariance estimation	$N_R^2\,(8BM\|S\|+2)$
Signal power estimation	$5N_{\text{FFT}}\log_2 N_{\text{FFT}} + 4N_T N_R N_{\text{FFT}}\|S\| + N_R$
Delay profile estimation	0 (can be merged to signal power estimation)
Doppler spectrum estimation	$5\|S\|^2 N_T N_R BM + 102\|S\|^2$
Robust channel estimation	$60(B^2M^2+\|S\|^2) + 6B^2M^2\|S\|^2 + (16B^3M^3\|S\|^3 + 8B^2M^2\|S\|^2 + 1 + BM\|S\|)N_T N_R$
Precoder, Rank selection	$(4N_T^2 N_R)BM\|S\| + \sum_{R=1}^{N_R}\left(8N_T^2R + 8R^2N_T + R + \frac{8}{3}R^3\right)O\binom{N_T}{R}$

number can be obtained by simply multiplying the number of FLOPs for the forward pass times 2).

The FLOPs for the self-attention mechanism are given by:

$$\text{FLOPs}_{\text{self-attention}} = 4\times\text{seq\_length}\times\text{token\_emb\_dim}^2\times\text{n\_heads}$$

The FLOPs for the feed-forward network (which has 2 layers) are given by:

$$\text{FLOPs}_{\text{feed-forward}} = 2\times\text{seq\_length}\times\text{token\_emb\_dim}\times\text{hidden\_dim}$$

The FLOPs for the embedding layer are given by:

$$\text{FLOPs}_{\text{embedding}} = \text{seq\_length} \times \text{token\_emb\_dim}$$

The total FLOPs per transformer layer is the sum of the FLOPs from the self-attention, and feed-forward:

$$\text{FLOPs}_{\text{layer}} = \text{FLOPs}_{\text{self-attention}} + \text{FLOPs}_{\text{feed-forward}}$$

The total FLOPs for the entire model is the product of the number of transformer layers and the FLOPs per transformer layer, summed with the FLOPs for the embedding layer:

$$\text{FLOPs}_{\text{model}} = \text{n\_layers} \times \text{FLOPs}_{\text{layer}} + \text{FLOPs}_{\text{embedding}}$$

The final number of FLOPs for training is then obtained by multiplying the cost of forward and backward pass with the number of training examples.

B. FLOPs for data generation

We list the FLOPs for each submodule in data generation in the Table VI. Since our dataset is generated uniformly over different configuration, we average the value of all settings as the FLOPs for generating one data point.