Clustering-Based Pilot Overhead Reduction for Channel Estimation in Dynamic Wireless MIMO Systems

1st Fayad Haddad, 2nd Lingrui Zhu, 3rd Carsten Bockelmann, 4th Armin Dekorsy
Department of Communications Engineering
University of Bremen, Germany
Email: {haddad, zhu, bockelmann, dekorsy}@ant.uni-bremen.de

***Abstract*—This paper proposes an adaptive framework for channel estimation that minimizes overhead by optimizing pilot design for dynamic wireless environments. Unlike traditional methods that often assume a static environment, this approach adapt to real-world systems where channel conditions vary significantly, leading to changes in the transmission channel model. As these fluctuations can significantly affect the performance of channel estimators, which are typically trained on a single, static channel model. The proposed framework offers a dynamic solution as it operates in two steps: first, it uses a reduced set of strategically placed pilots to identify the actual channel model, and then it applies a specialized, pre-trained network tailored for that specific model to perform channel estimation with fewer pilots. For reliable model identification, a pilot placement strategy using Concrete AutoEncoder (CAE) is employed. Once the channel model is identified, a Transformer-based network, fine-tuned to the detected model, performs the channel estimation by focusing on relevant model features, thus reducing pilot requirements. Simulation results show that this adaptive framework outperforms conventional methods that train a single estimator across various models, achieving more accurate channel estimation with lower pilot overhead.**

***Index Terms*—Channel estimation, dynamic wireless channels, MIMO systems, concrete autoencoder, transformer.**

I. Introduction

In wireless communication systems, the dynamic nature of the practical environment poses significant challenges. User mobility, moving obstacles, and varying scatterers lead to considerable changes in channel conditions, resulting in changes to the transmission channel model. This necessitates robust channel estimation systems that can adapt to diverse channel conditions. Traditional estimation methods, designed for static or slowly varying environments, often experience performance degradation when channel characteristics change significantly. Although improving channel estimation performance under variable conditions is possible, it typically requires an increased number of pilots, which can reduce spectral efficiency and ultimately lower data rates [1]. To address this challenge in highly dynamic environments, it is essential to develop strategies that effectively manage channel estimation amidst the variability of real-world conditions without imposing excessive pilot overhead.

Several schemes for channel estimation and reducing pilot overhead have been thoroughly explored. In [2], the authors leverage two intrinsic features of wireless channels, temporal correlation and time domain sparsity, to develop a channel estimator that combines Compressed Sensing (CS) with Dynamic Mode Decomposition (DMD). Since this method relies on temporal correlation, it cannot directly adapt to changes in the channel model. Additionally, it employs a random pilot placement due to its use of CS. Recently, Machine Learning (ML) has gained considerable attention for its effectiveness across various applications, including channel estimation [3]. Deep Learning (DL) networks show improved performance in challenging scenarios such as fading, signal distortion, and interference [1]. In [4], the authors propose a hybrid encoder-decoder architecture known as HA02, which incorporates a self-attention mechanism. The encoder is built on a state-of-the-art Transformer network that utilizes multi-head attention. This architecture integrates a Transformer encoder block with a residual neural network for the decoder.

However, even well-trained DL-based channel estimation networks may struggle with significant shifts in channel characteristics, as substantial environmental changes can modify the channel model [5], [6]. Consequently, the DL network must be trained on multiple channel models simultaneously, which can lead to reduced performance due to the decreased relevance of the training data to one another. Additionally, more pilots may be required to capture the increased number of features from the various channel models, needed to enable better differentiation between them.

In this paper, we present a novel adaptive framework for channel estimation that dynamically selects the most suitable channel model based on prevailing channel conditions. A key innovation is the design of a pilot pattern using a Concrete Autoencoder (CAE), which minimizes pilot overhead while effectively distinguishing between channel models. Based on the identified model, the framework selects a pre-trained channel estimation network optimized for that model, ensuring efficient and accurate estimation. By minimizing pilot usage and selecting a channel estimation network tailored to actual conditions, our framework can greatly enhance efficiency in dynamic wireless communication environments.

Notations: In this paper, matrices are denoted by uppercase boldface letters, column vectors by bold lowercase letters, scalars by italic lowercase letters, and numbering by italic uppercase letters. Hadamard product and division are indicated with $\odot$ and $\oslash$, respectively. $E\{.\}$ denotes the mean.

This work was funded by the German Ministry of Education and Research (BMBF) under grants 16KISK016 (Open6GHub) and 16KISK109 (6G-ANNA).

II. System and Channel Models

A. System Model

We consider a MIMO-Orthogonal Frequency Division Multiplexing (MIMO-OFDM) system with K subcarriers, N_t transmit antennas, and N_r receive antennas. The channel matrix $\mathbf{H} \in \mathbb{C}^{K\times N}$, where $N = N_t N_r$, represents the total MIMO channels. The transmitted signal $\mathbf{X} \in \mathbb{C}^{K\times N}$ includes pilot symbols and user data. The received signal $\mathbf{Y}$ is expressed accounting for the channel effect as:

$$\mathbf{Y} = \mathbf{H} \odot \mathbf{X} + \mathbf{Z}, \tag{1}$$

where $\mathbf{Z} \in \mathbb{C}^{K\times N}$ represents the additive noise with zero mean and variance σ_z^2 per element, respectively. To account for the channel noise, we introduce the signal-to-noise ratio (SNR), defined as $\frac{E\{\|\mathbf{X}\|_2^2\}}{\sigma_z^2}$, assuming pilots and data signals have equal power. The objective is to find the estimated channel $\hat{\mathbf{H}}$ using the pilots in $\mathbf{X}$. Due to the channel estimation error, $\hat{\mathbf{H}}$ can deviate from the actual $\mathbf{H}$. To evaluate performance, we employ the Normalized Mean Square Error (NMSE), defined as $\text{NMSE} = \frac{E\{\|\mathbf{H}-\hat{\mathbf{H}}\|_2^2\}}{E\{\|\mathbf{H}\|_2^2\}}$.

B. Channel Model

In wireless communication, due to the radio propagation environment, the transmitted signal reaches the receiver via multiple paths with different time delays and with different attenuation levels. For each antenna pair $n_t \in [1, 2, \ldots, N_t]$ and $n_r \in [1, 2, \ldots, N_r]$, the channel coefficient is defined according to Jake's model [7] as:

$$g_{n_t,n_r}(t) = \sum_{l=1}^{L} \beta_l \left(\zeta^{\text{LOS}} \psi_l^{\text{LOS}}(t) + \zeta^{\text{NLOS}} \psi_l^{\text{NLOS}}(t)\right), \tag{2}$$

where ζ^{LOS} and ζ^{NLOS} are the gains for the Line-Of-Sight (LOS) and Non-Line-Of-Sight (NLOS) components, respectively. For the lth path, the parameters of the power delay profile, (β_l, τ_l), represent the path gain and the time delay, respectively. ψ_l^{LOS} and ψ_l^{NLOS}, describe the fading effects for LOS and NLOS components and are expressed as:

$$\begin{aligned} \psi_l^{\text{LOS}} &= e^{j\left(2\pi f_D \cos(\theta_l^{\text{LOS}})t + \phi^{\text{LOS}}\right)}, \\ \psi_l^{\text{NLOS}} &= \frac{1}{\sqrt{N_{\text{sine}}}} \sum_{s=1}^{N_{\text{sine}}} e^{j\left(2\pi f_D \cos(\theta_{l,s}^{\text{NLOS}})(t-\tau_l) + \phi_s^{\text{NLOS}}\right)}, \end{aligned} \tag{3}$$

where f_D is the Doppler frequency, θ represents the angle of arrival, ϕ is the phase shift, and N_{sine} denotes the number of sine waves modeling the NLOS component. The term $\frac{1}{\sqrt{N_{\text{sine}}}}$ maintains the average power of the NLOS component across the summed signals. Using (2), the stochastic channel coefficients form the matrix $\mathbf{G}(t) \in \mathbb{C}^{N_t\times N_r}$ can be written as:

$$\mathbf{G}(t) = \begin{bmatrix} g_{1,1}(t) & \cdots & g_{1,N_r}(t) \\ \vdots & \ddots & \vdots \\ g_{N_t,1}(t) & \cdots & g_{N_t,N_r}(t) \end{bmatrix}. \tag{4}$$

To model the spatial correlation between antennas, the Kronecker model [8] is applied, as follows:

$$\tilde{\mathbf{G}}(t) = \mathbf{R}_t^{\frac{1}{2}} \mathbf{G}(t) \mathbf{R}_r^{\frac{1}{2}}, \tag{5}$$

where $\mathbf{R}_t \in \mathbb{C}^{N_t\times N_t}$ and $\mathbf{R}_r \in \mathbb{C}^{N_r\times N_r}$ are the spatial correlation matrices for the transmitter and receiver, respectively. To obtain the channel impulse response, we sample the MIMO channels $\tilde{\mathbf{G}}(t)$, resulting in $\tilde{\mathbf{G}} \in \mathbb{C}^{N_{\text{samples}}\times N_t\times N_r}$, as:

$$\tilde{\mathbf{G}} = \begin{bmatrix} \tilde{\mathbf{g}}_{1,1} & \cdots & \tilde{\mathbf{g}}_{1,N_r} \\ \vdots & \ddots & \vdots \\ \tilde{\mathbf{g}}_{N_t,1} & \cdots & \tilde{\mathbf{g}}_{N_t,N_r} \end{bmatrix}. \tag{6}$$

Next, we apply zero-padding to the first dimension to achieve a size of K, then reshape the result into a 2D matrix to obtain $\tilde{\mathbf{G}}' \in \mathbb{C}^{K\times N}$. Finally, we apply the Discrete Fourier Transform (DFT) to obtain $\mathbf{H} \in \mathbb{C}^{K\times N}$ as $\mathbf{H} = \text{DFT}(\tilde{\mathbf{G}}')$.

The power delay profile can be adjusted to reflect different environmental and mobility conditions [1]. 3GPP has standardized channel models, covering various environments and mobility scenarios [9]. In the following we provide a brief overview of the 3GPP-adopted channel models for 5G.

- **Cost_EPA:** Urban pedestrian model
- **Cost_ETU:** Urban model with high obstacles.
- **Cost_hilly:** Hilly terrain model with obstructed LOS Component.
- **Cost_rural:** Rural model with fewer obstacles and more open space.
- **Cost_urban:** Urban model with dense environment.
- **Cost_VehA:** Vehicular model with rapid environmental changes.

To visualize the distribution of coefficients from various channel models, we use t-SNE [10], a dimensionality reduction technique for revealing structure in complex datasets, such as identifying clusters. Fig. 1(a) shows distinct clusters formed for the EPA, Hilly, and ETU models in a noise-free scenario, clearly separating the models. However, as demonstrated in Fig. 1(b), introducing noise at an SNR of 3 dB causes the clusters to overlap, making the distribution indistinguishable. This highlights the challenges noise introduces in effectively classifying channel models.

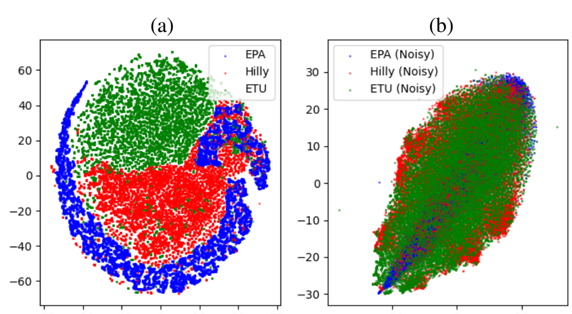

Fig. 1. t-SNE visualizations of different channel models. (a) Distribution without noise, showing distinct clustering. (b) Distribution with noise at SNR of value 3 dB, where clusters overlap.

III. Channel estimation and pilot design

In this section, we present the proposed technique for pilot design and channel estimation. Additionally, we discuss the conventional channel estimation techniques standardized by 3GPP for comparison purposes, namely Least Squares (LS) and Minimum Mean square Error (MMSE). Since the method is based using the CAE for pilot design and the HA02 network for channel estimation, we first introduce an overview of these neural networks then discuss the proposed method.

A. Pilot placement with concrete autoencoder

The concrete autoencoder [11] is a deep learning-based method for feature selection, which efficiently identifies a subset of the most informative features. It consists of a single concrete selector layer (encoding layer), and interpolation MultiLayer Perceptron (MLP) (decoding layers).

The Concrete Selector Layer is based on concrete random variables for continuous relaxation sampling [12]. The selector layer has P output neurons each of them is connected to all of the K input features. For each output node p, with $p \in \{1, \dots, P\}$, the input nodes are sampled based the parameters vector $\boldsymbol{\alpha}_p \in \mathbb{R}^K_{>0}$, with $\boldsymbol{\alpha}_p = [\alpha_{1,p}, \alpha_{2,p}, ..., \alpha_{K,p}]$, that initially specified randomly. The sampling weights are controlled by a temperature parameter $T \in (0, \infty)$ and based on the parameters $\boldsymbol{\alpha}$ as:

$$c_{k,p} = \frac{\exp((\log \alpha_{k,p} + u_{k,p})/T)}{\sum_{j=1}^{K} \exp((\log \alpha_{j,p} + u_{j,p})/T)}, \tag{7}$$

with u is randomly sampled from a Gumbel distribution. Each elements $c_{k,p}$ refers to the kth weight in the sample vector p, with $k \in \{1, \dots, K\}$. Consider the input of the selector layer is $\mathbf{d}_{\text{in}} \in \mathbb{R}^K$ and the output is $\mathbf{d}_{\text{out}} \in \mathbb{R}^P$ then it can be define $\mathbf{d}_{\text{out}} = \mathbf{C} \cdot \mathbf{d}_{\text{in}}$, where $\mathbf{C} \in \mathbb{R}^{P \times K}$ contains all elements $c_{k,p}$. This represents a weighted linear combination of the input features. At the start of training, the parameters in $\boldsymbol{\alpha}$ are initialized to small positive values to encourage the selector layer to explore different linear combinations of input features, while the temperature parameter T is set to a high value. As training progresses, T decreases towards zero and the weights become more sparse. Consequently, the concrete selector layer outputs exactly one input feature for each output node by the end of the training process. In our context, the positions of these selected input features are mapped to the pilot positions.

In [13], authors used the CAE to determine pilot positions for channel estimation through unsupervised learning. In our work, given a dynamic environment, we use the CAE to select pilot positions for channel classification. This enables the CAE to identify the most distinguishable features across channel models. A key difference is the use of supervised learning, with an adjusted decoder design. We also changed the loss function $\mathcal{L}$ to be *sparse categorical cross-entropy*, defined as: $\mathcal{L} = -\frac{1}{B}\sum_{i=1}^{B} \log(q_i)$, where q_i is the predicted probability of the true class and B is the Batch size. The decoder's output size corresponds to the number of channel models, with the highest index indicating the channel model label.

B. Transformer-Based HA02 Channel Estimator

The HA02 architecture, proposed in [4], features a hybrid structure that combines a transformer-based encoder with a residual convolutional-based decoder, as demonstrated in Fig, 2. To evaluate the network, we first extract pilot symbols from the true channel matrix $\mathbf{H}$, apply noise to them, and then resize to match $\mathbf{H}$ dimensions using linear interpolation, as in [14]. The real and imaginary parts are also separated and concatenated to form the encoder input.

HA02 Encoder: The transformer-based encoder utilizes a self-attention mechanism that allows the model to focus on the most relevant input features, improving channel estimation accuracy. Inputs to the multihead attention layer are generated from a linear transformation via a Fully Connected (FC) layer. This layer enables simultaneous attention to different parts of the input sequence, capturing various relationships. The Add & Norm step incorporates a residual connection to maintain input information and reduce internal covariate shift. The feed-forward network introduces non-linearity, refining the representation further.

HA02 Decoder The residual convolutional decoder architecture addresses the degradation problem by incorporating skip connections. It starts with a convolutional layer, followed by a residual convolutional block that includes two convolutional layers. An Add & Norm step follows, which features the skip connection from the output of the first convolutional layer, ensuring that crucial input information is retained. The decoder concludes with an upsampling section, comprising an FC layer and a final convolutional layer, which facilitates one-dimensional upsampling and enhances generalization across different SNR values.

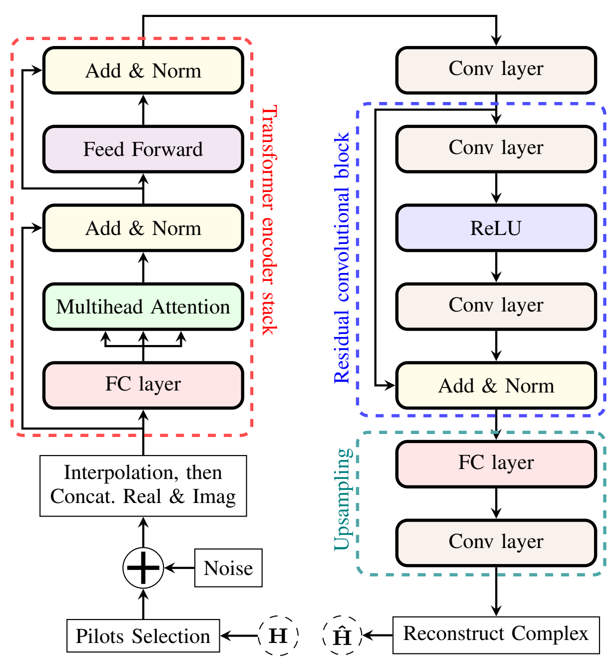

Fig. 2. Transformer-based HA02 architecture for channel estimation.

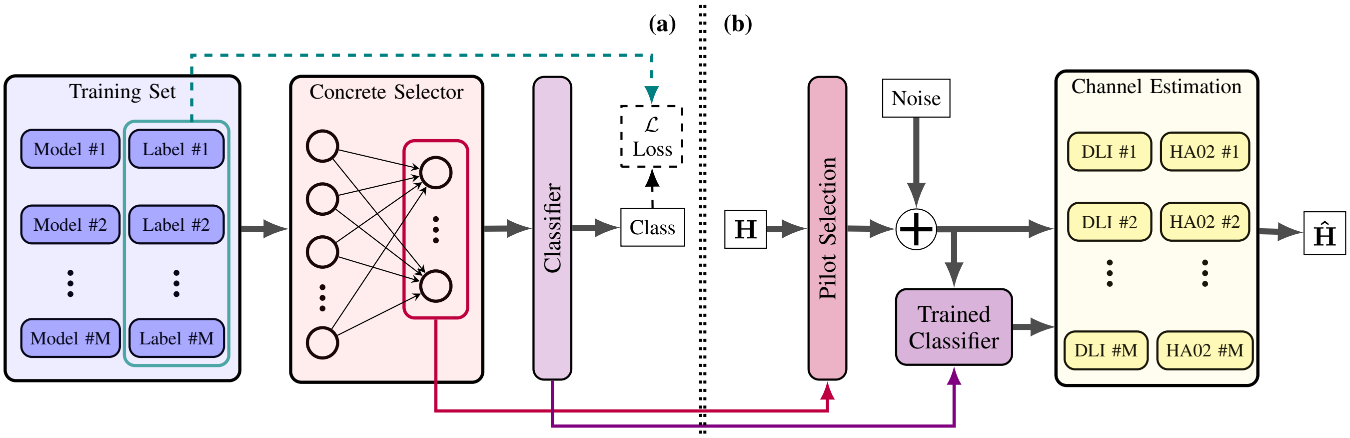

Fig. 3. The proposed framework, cluster-HA02. (a) illustrates the CAE-based classifier used to obtain the pilot positions tailored for channel model defining. (b) illustrates the selection of the DL-based Interpolation (DLI) and HA02 networks used for channel estimation based on the CAE-based pilots pattern.

C. Cluster-Based Channel Estimation

This section presents the integration of the CAE-based classifier with the HA02 channel estimation network, forming the cluster-HA02 method, as illustrated in Fig. 3.

The two-step strategy includes training the concrete selector layer with M predefined channel models (classes), where each model is associated with a unique class label, as illustrated in Fig. 3(a). Since the concrete selector processes input real values, we first take the absolute values from our training set before passing them through the concrete selector, which functions as the encoder for the CAE. For the decoder, we construct an MLP classifier, consisting of three FC layers, each followed by a ReLU activation function, and a final output FC layer is followed with a softmax activation. The output size of this classifier corresponds to M, with the maximum value's position indicating the class label.

The testing procedure of the proposed channel estimation technique is illustrated in Fig. 3(b), where all components of the framework are assumed to be pre-trained. Each transformer-based HA02 network is paired with a DL-based Interpolation (DLI) network. The DLI is an MLP-based network consists of a single FC layer with ReLU activation followed by another FC output layer. Among the M pairs of DLI-HA02 networks, each pair is pre-trained independently using a unique channel model, with the CAE-based pilot pattern taken into account. This pre-training follows the methodology outlined in [4]. During the testing phase, pilot symbols are selected from the channel matrix $\mathbf{H}$ based on the output of the concrete selector. These pilots are then corrupted by Additive White Gaussian Noise (AWGN), as described in Section II. The noisy pilots are then fed into the trained classifier to identify the corresponding channel model of the original channel $\mathbf{H}$. The resulting class information is used to select the appropriate pre-trained DLI-HA02 pair. The noisy pilots are also used as input into the selected DLI-HA02 network. This dual-purpose use of the pilots ensures efficient channel estimation with less pilot overhead.

D. Conventional Method for Channel Estimation

Conventional OFDM channel estimation methods are the Least Squares (LS) and Minimum Mean Square Error (MMSE), as introduced in [15]. The benchmarks implementation of LS and the MMSE are used for performance comparison with the proposed method.

Least Squares: The LS method aims to minimize the error between the transmitted and received signals over the known pilot symbols. Referring back to Section II, if we arrange the pilot symbols from the transmitted signal $\mathbf{X}$ and the received signal $\mathbf{Y}$ into matrices $\mathbf{X}_p$ and $\mathbf{Y}_p$, respectively, with dimensions $\mathbb{C}^{P\times N}$, then the LS approach is applied to estimate the channel coefficients specifically at the pilot positions by:

$$\hat{\mathbf{H}}_p^{\text{LS}} = \arg\min_{\mathbf{H}_p} \|\mathbf{Y}_p - \mathbf{H}_p \odot \mathbf{X}_p\|_2^2 = \mathbf{Y}_p \oslash \mathbf{X}_p, \tag{8}$$

with $\hat{\mathbf{H}}_p^{\text{LS}}$ and $\mathbf{H}_p \in \mathbb{C}^{P\times N}$ represent the estimated and actual channel coefficients at the pilot positions, respectively. LS is considered a straightforward low complexity method. To find the estimated channel matrix $\hat{\mathbf{H}}^{LS}$, we resize $\hat{\mathbf{H}}_p^{\text{LS}}$ by performing linear interpolation as $\hat{\mathbf{H}}^{\text{LS}} = \mathbf{A}_{\text{LI}} \cdot \hat{\mathbf{H}}_p^{LS}$, with $\mathbf{A}_{\text{LI}} \in \mathbb{C}^{K\times P}$ the linear interpolation matrix [14].

Minimum Mean Square Error: MMSE method can be utilized generally to estimate the channel matrix by minimizing the mean square error between the actual channel and the estimated channel $\hat{\mathbf{H}}$:

$$\hat{\mathbf{H}}^{\text{MMSE}} = \arg\min_{\hat{\mathbf{H}}} \mathbb{E}\left\{\|\mathbf{H} - \hat{\mathbf{H}}\|_2^2\right\}. \tag{9}$$

Here we use linear estimator to estimate channel matrix according to LS-based estimation obtained in (8), defined as:

$$\hat{\mathbf{H}}^{\text{MMSE}} = \mathbf{A}^{\text{MMSE}} \cdot \hat{\mathbf{H}}_p^{\text{LS}}. \tag{10}$$

With linear estimator, the MMSE problem can be converted to a Linear-MMSE problem (LMMSE), described as:

$$\begin{aligned} \mathbf{A}^{\text{MMSE}} &= \arg\min_{\mathbf{A}_{\text{LI}}} \mathbb{E}\left\{\|\mathbf{H} - \mathbf{A}_{\text{LI}} \cdot \hat{\mathbf{H}}_p^{\text{LS}}\|^2\right\}, \\ &= \mathbf{R}_{hp}\left(\mathbf{R}_{pp} + \sigma_z^2 \mathbf{I}\right)^{-1}, \end{aligned} \tag{11}$$

where $\mathbf{R}_{hp} \in \mathbb{C}^{K\times P}$ is the cross correlation matrix between actual channel matrix to be estimated and pilot matrix $\mathbf{H}_p$. $\mathbf{R}_{pp} \in \mathbb{C}^{P\times P}$ is the autocorrelation matrix of $\mathbf{H}_p$. According to (11), the MMSE estimation can be calculated as:

$$\hat{\mathbf{H}}^{\text{MMSE}} = \mathbf{R}_{hp}\left(\mathbf{R}_{pp} + \sigma_w^2 \mathbf{I}\right)^{-1} \cdot \hat{\mathbf{H}}_p^{\text{LS}}. \tag{12}$$

The LS and MMSE methods are widely recognized in wireless communication systems, including those standardized by 3GPP, and serve as benchmarks for performance comparison. Their effectiveness in channel estimation within OFDM systems makes them ideal for evaluating the improvements of the proposed method.

IV. Simulation Setup and Results

In this section, we perform numerical simulations to evaluate the performance of the proposed channel estimation method, cluster-HA02, and compare the performance for different scenarios.

A. Simulations Setup

We employ Heterogenous Radio Mobile Simulator (HermesPy) [16] to generate the channel coefficients of the mentioned channel models in Section II. The system parameters used are listed in Table I.

TABLE I
Simulation Parameters

System Parameters	Value
Carrier frequency f_c	2.5 GHz
No. of BS antenna N_t	4
No. of MS antenna N_r	4
Subcarrier spacing	15 KHz
No. of subcarriers K	256
Channel estimating error	AWGN

To train the proposed framework, we utilize training sets specific to each channel model, with each set containing 10000 realizations. The training process utilizes the Adam optimizer across all discussed networks. Further training parameters are depicted in Table II

TABLE II
Training Parameters

	CAE	DLI	HA02
Batch size	40	100	50
No. of Epochs	100	100	100
Learning rate	0.001	0.001	0.002

B. Simulation Results

First, we assess the performance of the CAE-based classifier across different numbers of channel models (classes) M, starting with $M = 3$ and simulating up to $M = 6$. The selected channel models follow the arrangement as in Section II.
In Fig. 4(a), we present the classification accuracy for varying M as a function of the number of pilots P, assuming perfect channel estimation on the pilots positions. The results indicate that classification accuracy improves with an increased number of pilots, though this improvement is nonlinear and tends to plateau at higher P. This behavior is expected, as more pilots capture additional information about the classes, facilitating better differentiation. Additionally, reducing M generally enhances accuracy since the classifier faces fewer categories to distinguish, thereby decreasing the likelihood of misclassification. With fewer classes, there is typically less inter-class similarity, which further aids the classifier in effectively distinguishing the classes and boosting overall accuracy.

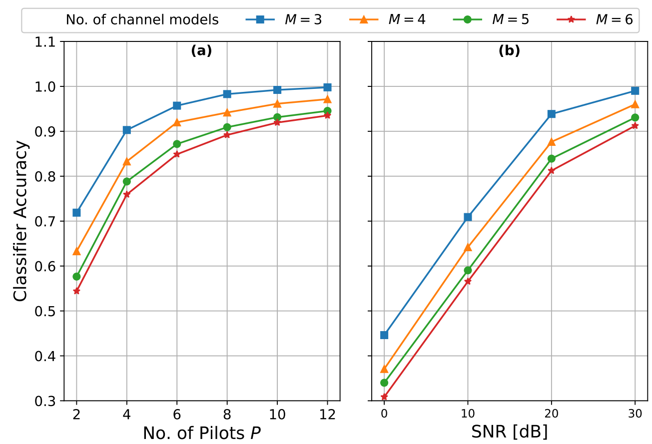

Fig. 4. Comparison of classifier accuracy for different number of channel models M (classes). (a) in terms of number of pilots P with perfect channel estimation. (b) in terms of channel estimation SNR with $P = 12$.

Fig. 4(b) illustrates the classifier's accuracy performance in relation to channel estimation error presented as SNR, using $P = 12$. The simulation results reveal that higher SNR values result in improved accuracy. Noise negatively impacts classifier performance by distorting pilot signals and obscuring essential features required for identification, leading to increased misclassification. This result aligns with Fig. 1, where it is evident that the added noise can render the channel models indistinguishable.

Now, we evaluate the performance of the proposed cluster-HA02 method compared to the standard HA02 without clustering across two different scenarios, where the number of channel models is set to $M = 3$ and $M = 6$. The performance of each method is evaluated against traditional benchmarks LS and MMSE channel estimators, with their results averaged over samples from all six available channel models. In all cases, the number of pilots remains fixed at $P = 12$, ensuring a fair comparison. For HA02, LS, and MMSE, the pilot positions are evenly spaced, adhering to their design in the original algorithms. In contrast, cluster-HA02 leverages a CAE-based classifier to determine the optimal pilot positions, tailored specifically to improve the model's clustering. It is worth mentioning that the original HA02 work employed bilinear interpolation. However, we utilize the simpler linear interpolation, as discussed in Section III-B.

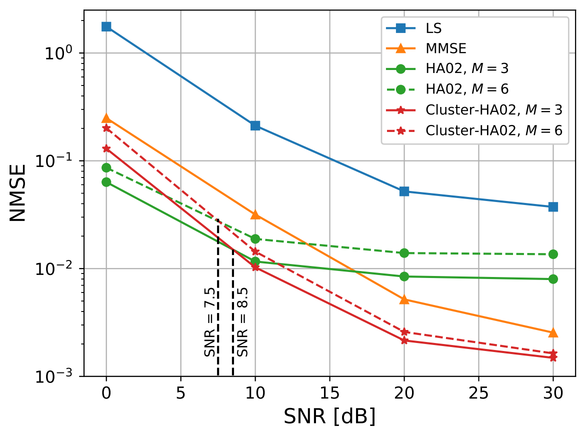

Fig. 5. NMSE performance for channel estimation in terms of SNR with $P = 12$. The HA02 and cluster-HA02 are trained for $M = 3$ and for $M = 6$. The LS and MMSE results are averaged over all 6 channel models.

Simulation results in Fig. 5 reveal that incorporating classification improves system performance under certain conditions, as indicated by a reduction in NMSE relative to the conventional HA02 approach. For the case of $M = 3$, cluster-HA02 outperforms HA02 when the SNR exceeds 8.5 dB, demonstrating that higher SNR values correspond to enhanced performance for cluster-HA02. Conversely, for SNR values below 8.5 dB, HA02 exhibits superior performance compared to cluster-HA02. Similar trends are observed for $M = 6$, although the overall performance is somewhat diminished due to the decreased classification accuracy associated with the increased number of classes, as detailed in Fig. 4. Notably, increasing M leads to a more pronounced performance drop for HA02, particularly at moderate SNR values. This suggests that training HA02 across multiple channel models simultaneously introduces higher confusion to the network compared to training the CAE-based classifier on the same dataset within the cluster-HA02 framework.

Generally, the slope of the cluster-HA02 curve is steeper than that of HA02 alone, highlighting the impact of channel estimation noise on the classification accuracy, as shown in Figure 4(b). At low SNR levels, the classifier is more likely to misidentify the channel model, resulting in a significant drop in performance. In contrast, at higher SNR levels, the classifier's accuracy improves substantially, facilitating better channel estimation as the selected channel estimator aligns closely with the corresponding channel model.

V. Conclusion

This paper introduces a novel adaptive framework for channel estimation, designed to address the challenge of pilot overhead in dynamic wireless environments. The method employs optimized pilot patterns tailored for effective channel model identification. The proposed two-step strategy, combining targeted pilot placement with model-specific estimation, demonstrates improved channel estimation performance compared to approaches that assume a static channel model for the same number of pilots. This approach can reduce the number of pilots required to achieve a desired estimation accuracy.

By leveraging a Concrete AutoEncoder for pilot design, the framework enables efficient channel model identification in favorable SNR conditions, allowing the subsequent Transformer-based HA02 network to focus on model-specific features. This makes the approach particularly suitable for moderate SNR levels and manageable numbers of channel models. However, in scenarios with a high number of potential channel models that a user may encounter, the performance of HA02-based estimation may be limited, as shown by our findings.

Overall, this tailored approach not only enhances channel estimation efficiency in dynamic environments, but also consistently outperforms traditional methods relying on static pilot design. Simulation results validate the framework's potential for optimizing channel estimation in dynamic, real-world environments, requiring less pilots to achieve a target estimation performance.

References

[1] T. S. Rappaport, *Wireless communications: principles and practice*. Cambridge University Press, 2024.

[2] F. Haddad, C. Bockelmann, and A. Dekorsy, "Channel estimation and pilot overhead reduction in ofdm systems using compressed sensing dynamic mode decomposition," *IEEE Communications Letters*, 2024.

[3] Y. C. Eldar, A. Goldsmith, D. Gündüz, and H. V. Poor, *Machine learning and wireless communications*. Cambridge University Press, 2022.

[4] D. Luan and J. Thompson, "Attention based neural networks for wireless channel estimation," in *2022 IEEE 95th Vehicular Technology Conference:(VTC2022-Spring)*, pp. 1–5, IEEE, 2022.

[5] L. Dai, R. Jiao, F. Adachi, H. V. Poor, and L. Hanzo, "Deep learning for wireless communications: An emerging interdisciplinary paradigm," *IEEE Wireless Communications*, vol. 27, no. 4, pp. 133–139, 2020.

[6] Z. Chen, Z. Zhang, and Z. Yang, "Big ai models for 6g wireless networks: Opportunities, challenges, and research directions," *IEEE Wireless Communications*, 2024.

[7] U. Karabulut, A. Awada, I. Viering, A. N. Barreto, and G. P. Fettweis, "Low complexity channel model for mobility investigations in 5g networks," in *2020 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–8, IEEE, 2020.

[8] C. Oestges, "Validity of the kronecker model for mimo correlated channels," in *2006 IEEE 63rd Vehicular Technology Conference*, vol. 6, pp. 2818–2822, IEEE, 2006.

[9] "ETSI TR 13 8901." [Online]. Available: https://www.etsi.org/deliver/etsi_tr/138900_138999/138901/16.01.00_60/tr_138901v160100p.pdf.

[10] L. van der Maaten and G. Hinton, "Visualizing data using t-sne. journal of machine learning research 9," *Nov (2008)*, 2008.

[11] M. F. Balın, A. Abid, and J. Zou, "Concrete autoencoders: Differentiable feature selection and reconstruction," in *International conference on machine learning*, pp. 444–453, PMLR, 2019.

[12] C. J. Maddison, A. Mnih, and Y. W. Teh, "The concrete distribution: A continuous relaxation of discrete random variables," *arXiv preprint arXiv:1611.00712*, 2016.

[13] M. Soltani, V. Pourahmadi, and H. Sheikhzadeh, "Pilot pattern design for deep learning-based channel estimation in ofdm systems," *IEEE Wireless Communications Letters*, vol. 9, no. 12, pp. 2173–2176, 2020.

[14] X. Dong, W.-S. Lu, and A. C. Soong, "Linear interpolation in pilot symbol assisted channel estimation for ofdm," *IEEE transactions on wireless communications*, vol. 6, no. 5, pp. 1910–1920, 2007.

[15] R. Srividhya, T. Jayachandran, V. Rajmohan, *et al.*, "Channel estimation for ofdm systems using mmse and ls algorithms," in *2022 6th international conference on trends in electronics and informatics (ICOEI)*, pp. 1–5, IEEE, 2022.

[16] "HermesPy." [Online]. Available: https://www.barkhauseninstitut.org/en/results/hermespy.