Continual Learning for Wireless Channel Prediction

Muhammad Ahmed Mohsin [*1] **Muhammad Umer** [*1] **Ahsan Bilal** [*2] **Muhammad Ali Jameshed** [3]
John M. Cioffi [1]

Abstract

Modern 5G/6G deployments routinely face *cross-configuration handovers*—users traversing cells with different antenna layouts, carrier frequencies, and scattering statistics—which inflate channel-prediction NMSE by **37.5%** on average when models are naively fine-tuned. The proposed improvement frames this mismatch as a continual-learning problem and benchmarks three adaptation families: replay with loss-aware reservoirs, synaptic-importance regularization, and memory-free learning-without-forgetting. Across three representative 3GPP urban micro scenarios, the best replay and regularization schemes cut the high-SNR error floor by up to **2 dB** (≈35%), while even the lightweight distillation recovers up to 30% improvement over baseline handover prediction schemes. These results show that targeted rehearsal and parameter anchoring are essential for handover-robust CSI prediction and suggest a clear migration path for embedding continual-learning hooks into current channel prediction efforts in 3GPP—NR (Polese et al., 2018) and O-RAN (Garcia-Saavedra & Costa-Perez, 2021). The full codebase can be found at github.com/ahmd-mohsin/cl-wcp.

1. Introduction

Channel state information (CSI) prediction and estimation are long-unsolved problems in wireless communications and a bottleneck to many advanced physical-layer designs. Accurate CSI is essential for multi-antenna systems, but rapid channel variations ("channel aging") make timely CSI acquisition difficult in practice. 5G network radio (NR) specifications (e.g. TDD mode) mandate uplink sounding only every $\geq$ 2ms (Villena-Rodriguez et al., 2024). For instance, a 28GHz link with a 60km/h user has a coherence time on the order of 0.3ms (Villena-Rodriguez et al., 2024), so the channel may drift significantly between pilots. Consequently, outdated CSI can severely degrade throughput: studies show that even a 4ms feedback delay at moderate speed (30km/h) can reduce sum-rate by ≈50% at a carrier frequency of 3.5 GHz (Li et al., 2021). In practice, this implies that at higher speeds (e.g. 60km/h) the achievable rate can drop by tens of percent if prediction fails. These facts underscore the critical need for advanced channel prediction to preempt channel aging under 3GPP timing constraints.

Traditional statistical or model-based predictors cannot fully capture real-world channel dynamics, so learning-based methods have been explored (Jiang & Schotten, 2019). Recurrent neural networks (RNNs) with Long short-term memory (LSTM)/gated recurrent units (GRU) (Greff et al., 2016), (Dey & Salem, 2017) and attention-based models (Transformers) can learn to forecast channel time-series from past CSI (Joo et al., 2019). However, these data-driven predictors work well only when test channels closely match training conditions. In fact, deep predictors "exhibit poor generalization, requiring retraining when the CSI distribution changes" (Liu et al., 2024). The mismatch in array geometry or mobility can lead to large errors: for example, LSTM/GRU models incur a prediction error of 37.5% when moving from standard to dense environments (detailed description in Section B of the Appendix). Similarly, changing antenna spacing or carrier frequency can worsen the NMSE by on the order of 15-30% over the nominal case. A simple change of antenna tilt, array spacing, and polarization between different network conditions causes a prediction error of 34% as Figure 2 shows (see Appendix A and Figure 4 for the full baseline distributions and zero-shot evaluation). Moreover, sequential multi-step forecasting suffers from error accumulation; small mistakes compound over time, and naïve fine-tuning on new data causes catastrophic forgetting of previous channel patterns. In short, standard deep

[*]Equal contribution [1]Department of Electrical Engineering, Stanford University, Stanford, CA 94305, USA [2]School of Computer Science, University of Oklahoma, Norman, OK 73019, USA [3]School of Engineering, University of Glasgow, G12 8QQ, Glasgow, UK. Correspondence to: Muhammad Ahmed Mohsin <muahmed@stanford.edu>, Muhammad Umer <mumer@stanford.edu>, Ahsan Bilal <ahsan.bilal-1@ou.edu>, Muhammad Ali Jameshed <muhammadali.jamshed@glasgow.ac.uk>, John M. Cioffi <cioffi@stanford.edu>.

Proceedings of the 42nd International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025. Copyright 2025 by the author(s).

learning predictors lack cross-configuration generalization, and their sequential operation can amplify errors. This motivates the need for continual adaptation (Kirkpatrick et al., 2017). The proposed method incorporates Learning without Forgetting (LwF) (Li & Hoiem, 2017) style distillation to preserve older outputs while training on new data.

Contributions. To address these challenges, we propose a continual learning framework as shown in Figure 1 that incrementally adapts the channel predictor to evolving network conditions without catastrophic forgetting. This approach integrates replay and regularization techniques while maintaining a buffer of past channel examples for experience replay (Rolnick et al., 2019). It also applies methods like Elastic Weight Consolidation (EWC) (Zhou et al., 2022) and Synaptic Intelligence (Zenke et al., 2017) to penalize changes to weights important for old channels. Empirical evaluation shows that Experience Replay (ER), Loss Regularization Method (LRM), and LwF reduce cross-domain channel prediction error by up to 35%, 32%, and 25%, respectively, across diverse network configurations. Furthermore, training a single model continuously over all domains yields an additional 10% improvement in within-domain prediction accuracy. By incorporating loss-aware experience replay into ER and synaptic intelligence into LRM, we achieve a further average gain of 10% across datasets. Collectively, this novel continual learning framework delivers robust generalization under cross-network configurations.

Related Work. Data-driven CSI prediction has attracted considerable interest. (Liu et al., 2019) proposes an LSTM-based vehicular channel predictor capturing temporal dynamics to outperform ARIMA under a specified velocity profile. (Jiang & Schotten, 2020) introduced an LSTM predictor for fading channels, exploiting memory for multi-tap channel dependencies, although RNNs suffer from vanishing gradients and limiting lookaheads. (Jiang et al., 2022) proposed a parallel-attention scheme (JSAC 2022) that forecasts multiple future frames simultaneously. This transformer-based predictor largely eliminates "mobility-induced" error but demands extensive matched-condition training and high computational complexity. (Zhang et al., 2024) extended the attention mechanism to vehicular links: in an RSMA-enabled V2X system they use multi-head attention across subcarriers for CSI prediction, yielding higher rates but at the cost of large model size and training effort. (Liu et al., 2024) explored foundation models to adapt a GPT-2 language model to MISO-OFDM prediction, introducing pre-trained weight sharing for CSI. This enables zero-shot use with minimal fine-tuning but suffers from a natural language and CSI domain gap.

The attention mechanism has been widely used to capture long-range temporal correlation in CSI sequences. (Kim et al., 2025) has explored self-attention to weigh the relevance of different historical pilots when predicting the future channel. Such transformer-based methods can mitigate error propagation over one or two steps, but they still face challenges for long-horizon forecasts. Data-driven channel predictors are known to "encounter difficulties when dealing with unseen channel conditions" (Kim et al., 2025). In concrete terms, a transformer trained on one antenna array will typically see 15-30% worse NMSE when used on a different array spacing (see Figure 4 in Appendix). Domain shift forces SOTA attention models to be retrained or adapted for new configurations, restricting their out-of-the-box applicability.

Generative models—particularly diffusion models—have recently appeared in wireless channel research, mostly for CSI synthesis and augmentation. (Lee et al., 2024) condition a diffusion model on UE position to sample new MIMO channel matrices, effectively augmenting a small measured dataset. These synthetic channels can improve tasks like CSI compression or beam selection, as demonstrated in the paper. (Bhattacharya et al., 2025; Zilberstein et al., 2024) explore Langevin dynamics for joint source channel estimation under MIMO scenarios but face computational time inefficiency. Existing models omit online drift adaptation, motivating our continual learning approach to MIMO channel prediction.

2. System Model

2.1. Dataset

The dataset is synthesized with QuaDRiGa (Jaeckel et al., 2014), fixing the carrier at 5 GHz, the bandwidth at 100 MHz, and sampling 500 time instants across 18 OFDM resource blocks for each run. The details for the urban micro channel (UMi) are provided in Section B of the Appendix (see Table 2 for parameter settings and Figure 5 for channel-gain distributions). For every Monte-Carlo seed, 256 users have randomly chosen azimuths and configuration-dependent ranges; then they initialize linear tracks that match QuaDRiGa's sample-density constraints. To overcome the simulator's seeding artifacts and expose temporal correlation, users are displaced slightly at each iteration so successive channels remain correlated sand the predictor must infer the evolution to the next state.

2.2. Model Configuration

For continual-learning channel prediction, we test 3 different machine learning models to evaluate the robustness of our proposed pipeline and achieve accurate results. LSTM (Greff et al., 2016) performs the best overall across all datasets. Gated recurrent unit (GRU) (Dey & Salem, 2017) and Transformer (Han et al., 2021) both perform equally well under different scenarios. Due to space constraints, complete ar-

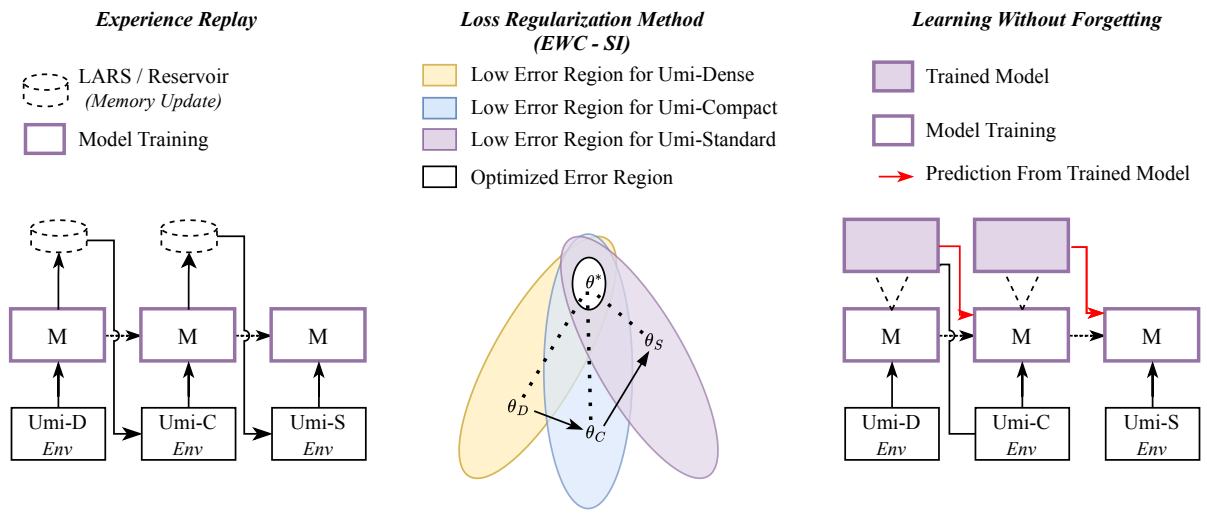


Figure 1. Algorithmic flow for continual learning under data drift mismatch for MIMO channel prediction.

chitectural diagrams and hyperparameter settings for the LSTM, GRU and Transformer backbones appear in Appendix A.

3. Continual Learning for Channel Prediction

Dynamic wireless channels induce distribution shift (*data drift*; see Section 1), so a predictor trained once quickly becomes stale across network configurations. Continual learning offers a remedy: given a sequence of datasets $\mathcal{D}_1, \mathcal{D}_2, \ldots$, the model updates on $\mathcal{D}_k$ while regularizing against weight changes that would degrade performance on $\bigcup_{i<k} \mathcal{D}_i$. Without such constraints, naïve sequential training biases the weights toward the most recent distribution, causing catastrophic forgetting on earlier conditions and yielding sub-optimal performance overall.

3.1. Experience Replay

ER is a powerful tool in continual learning where on-policy learning from novel instances (current dataset) and off-policy learning (Maei et al., 2010) from replay experiences (previous dataset) are interleaved. ER replays past channel samples to mitigate forgetting under dynamic wireless configurations. This reduces catastrophic forgetting (Kemker et al., 2018) and allows the same channel prediction models to be continuously used on real-time datasets gathered from different environments for better efficacy.

A replay buffer, $\mathcal{M}$, with a defined maximum capacity N_{buffer}, serves as the repository for these experiences. When a new experience e_t occurs at time t (corresponding to a new data sample from a specific network configuration), a decision is made regarding its inclusion in $\mathcal{M}$ based on a predetermined strategy. A user equipment (UE) that roams across several cells experiences *task shifts* in the underlying propagation environment—e.g. UMi-*compact* $\rightarrow$ UMi-*dense* $\rightarrow$ UMi-*standard*, where UMi stands for Urban microcell. To maintain reliable link adaptation, the channel-state predictor must acquire the current cell's statistics *without catastrophically forgetting* the statistics learned in previous cells' network configurations. We follow the continual-learning paradigm of *experience replay* to achieve this trade-off (Liu et al., 2023).

Replay buffer. Algorithm 1 allocates a fixed-size memory $\mathcal{M} = \{e^{(n)}\}_{n=1}^{N_{\text{buffer}}}$ that stores past observations, $e^{(n)} = \left(\mathbf{X}^{(n)}, \mathbf{H}^{(n)}, \phi^{(n)}\right)$ where $\mathbf{X}^{(n)} \in \mathbb{C}^{2\times T\times N_{\text{tx}}\times N_{\text{rx}}}$ is a sequence of T past channel realizations (real/imag split) and $\mathbf{H}^{(n)}$ is the target channel matrix to be predicted. Throughout training we denote by $\mathcal{D}_k$ the mini-dataset collected in the k-th cell.

Mini-batch composition. Global training step t (UE currently in cell k) draws a mini-batch (Krutsylo, 2024) $\mathcal{B}_t = \mathcal{B}_{\text{current}} \cup \mathcal{B}_{\text{replay}}$ where $\mathcal{B}_{\text{current}} \subset \mathcal{D}_k$ and $\mathcal{B}_{\text{replay}} \subset \mathcal{M}$. Elements of $\mathcal{B}_{\text{replay}}$ act as *rehearsal anchors* that remind the network of previous propagation settings. Mathematically, the training process with experience replay can be formulated as follows. Let $\boldsymbol{\theta}$ represent the parameters of the channel prediction model. The loss function associated with the current task, $\mathcal{L}_{\text{current}}(\boldsymbol{\theta}, \mathcal{B}_{\text{current}})$, is computed based on the model's predictions on the data from the current network configuration. When experience replay is employed, a mini-batch of past experiences, $\mathcal{B}_{\text{replay}}$, is sampled from the replay buffer $\mathcal{M}$. A loss term, $\mathcal{L}_{\text{replay}}(\boldsymbol{\theta}, \mathcal{B}_{\text{replay}})$, is then calculated based on the model's performance on this replayed data.

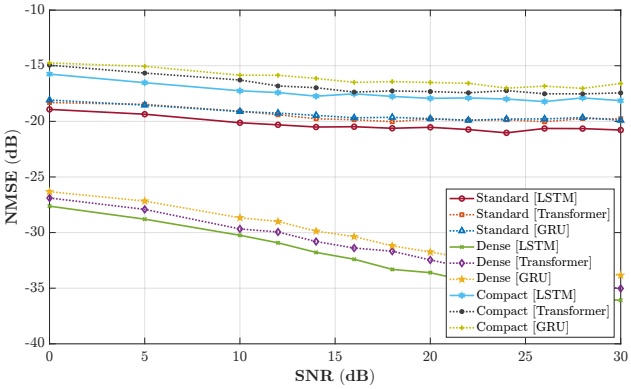


Figure 2. Baseline SNR trained on UMi dense and tested on all scenarios under all architectures.

The overall loss function for a given training step can be a weighted combination of these two loss terms (Fujimoto et al., 2020):

$$\mathcal{L}_{\text{total}}(\boldsymbol{\theta}) = \lambda \mathcal{L}_{\text{current}}(\boldsymbol{\theta}, \mathcal{B}_{\text{current}}) + (1-\lambda)\mathcal{L}_{\text{replay}}(\boldsymbol{\theta}, \mathcal{B}_{\text{replay}}) \tag{1}$$

where $\lambda \in [0, 1]$ is a hyperparameter that governs the balance between learning from the current task and rehearsing past experiences. Channel prediction specifically employs the normalized mean square error (NMSE) as the loss function. For the current dataset, this is:

$$\mathcal{L}_{\text{current}}(\boldsymbol{\theta}, \mathcal{B}_{\text{current}}) = \frac{1}{|\mathcal{B}_{\text{current}}|} \sum_{i=1}^{|\mathcal{B}_{\text{current}}|} \frac{\|\mathbf{H}_i - \hat{\mathbf{H}}_i(\boldsymbol{\theta})\|_F^2}{\|\mathbf{H}_i\|_F^2} \tag{2}$$

where $\mathbf{H}_i$ represents the true channel matrix for the i-th sample in $\mathcal{B}_{\text{current}}$, $\hat{\mathbf{H}}_i(\boldsymbol{\theta})$ is the predicted channel matrix using parameters $\boldsymbol{\theta}$, and $\|\cdot\|_F$ denotes the Frobenius norm. Similarly, for the replayed experiences, the loss is:

$$\mathcal{L}_{\text{replay}}(\boldsymbol{\theta}, \mathcal{B}_{\text{replay}}) = \frac{1}{|\mathcal{B}_{\text{replay}}|} \sum_{j=1}^{|\mathcal{B}_{\text{replay}}|} \frac{\|\mathbf{H}_j - \hat{\mathbf{H}}_j(\boldsymbol{\theta})\|_F^2}{\|\mathbf{H}_j\|_F^2} \tag{3}$$

The complete training objective with experience replay thus becomes:

$$\mathcal{L}_{\text{total}}(\boldsymbol{\theta}) = \lambda \left(\frac{1}{|\mathcal{B}_{\text{current}}|} \sum_{i=1}^{|\mathcal{B}_{\text{current}}|} \frac{\|\mathbf{H}_i - \hat{\mathbf{H}}_i(\boldsymbol{\theta})\|_F^2}{\|\mathbf{H}_i\|_F^2} \right) \tag{4}$$

$$+ (1-\lambda) \left(\frac{1}{|\mathcal{B}_{\text{replay}}|} \sum_{j=1}^{|\mathcal{B}_{\text{replay}}|} \frac{\|\mathbf{H}_j - \hat{\mathbf{H}}_j(\boldsymbol{\theta})\|_F^2}{\|\mathbf{H}_j\|_F^2} \right) \tag{5}$$

The mixing ratio λ is another crucial hyperparameter (Fedus et al., 2020). Larger λ biases optimization toward the current configuration's loss, driving parameters away from prior-environment optima and thus causing catastrophic forgetting. Conversely, a lower value of λ places more emphasis on the replayed data, encouraging the model to retain knowledge from past tasks but potentially hindering its ability to learn new patterns from the current data. In this 3-task study (UMi : {compact, dense, standard}) we fix $N_{\text{buf}} = 5000$ samples $\approx$ 10MB, small enough to be cached on the gNB side. Furthermore, high λ expedites adaptation after handover but risks NMSE spikes when the UE returns to a previous cell; low λ yields smoother performance across cells at the cost of slower convergence.

Algorithm 1 Continual Channel Prediction with Experience Replay (Reservoir – LARS)

Input: buffer size N_{buf}, mixing weight λ, sampling mode $s \in \{\text{UNIFORM}, \text{LARS}\}$, learning rate η
Initialize replay buffer $\mathcal{M} \leftarrow \emptyset$, counter $t \leftarrow 0$, model parameters θ
function `Insert`($\mathbf{X}$, $\mathbf{H}$, ℓ)
  $t \leftarrow t + 1$
  if $|\mathcal{M}| < N_{\text{buf}}$ **then**
    $\mathcal{M} \leftarrow \mathcal{M} \cup \{(\mathbf{X}, \mathbf{H}, \ell)\}$
  else if rand() $< \frac{N_{\text{buf}}}{t}$ **then**
    if $s = \text{LARS}$ **then**
      choose victim v using Eq. (7)
    else
      $v \leftarrow \text{randint}(1, |\mathcal{M}|)$
    end if
    $\mathcal{M}[v] \leftarrow (\mathbf{X}, \mathbf{H}, \ell)$
  end if
end function
for each visited cell k **do do**
  for each measurement $(\mathbf{X}_t, \mathbf{H}_t)$ **do do**
    $\hat{\mathbf{H}}_t \leftarrow f_\theta(\mathbf{X}_t)$
    $\ell_t \leftarrow \ell_{\text{NMSE}}(\mathbf{H}_t, \hat{\mathbf{H}}_t)$
    `Insert`($\mathbf{X}_t$, $\mathbf{H}_t$, ℓ_t)
    if ready to update **then**
      sample $\mathcal{B}_{\text{curr}} \subset \mathcal{D}_k$, $\mathcal{B}_{\text{rep}} \subset \mathcal{M}$
      compute $\mathcal{L}_{\text{total}}$ via Eq. (4)
      $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}_{\text{total}}$
    end if
  end for
end for

Reservoir Sampling. In the continual-learning pipeline, the UE encounters a *stream* of channel measurements while traversing the UMi-{compact, dense, standard} layouts. At step t Algorithm 1 observes the pair $e_t = (\mathbf{X}_t, \mathbf{H}_t)$, where $\mathbf{X}_t \in \mathbb{C}^{2\times T\times N_{\text{tx}}\times N_{\text{rx}}}$ is the window of the *past* T channel snapshots (real/imaginary split) and $\mathbf{H}_t \in \mathbb{C}^{2\times N_{\text{tx}}\times N_{\text{rx}}}$ is the *next-slot* channel to be predicted. We maintain a replay buffer $\mathcal{M}$ of fixed cardinality N_{buffer}, physically cachable at the serving gNB (5G base station) (Kim et al., 2020). To

Algorithm 2 Continual Channel Prediction with EWC

Input: learning rate η, stability coefficient α
Initialize model parameters θ
Initialize bank of snapshots $\mathcal{B} \leftarrow \emptyset$ {stores (θ_j^*, F_j)}
for each task D_k (*UMi-compact* $\rightarrow$ *dense* $\rightarrow$ *standard*) **do do**
  for each mini-batch $(\mathbf{X}, \mathbf{H}) \in D_k$ **do do**
    $\hat{\mathbf{H}} \leftarrow f_\theta(\mathbf{X})$
    $\ell_{\text{NMSE}} \leftarrow \ell_{\text{NMSE}}(\mathbf{H}, \hat{\mathbf{H}})$
    $\ell_{\text{EWC}} \leftarrow \frac{\alpha}{2} \sum_{(\theta_j^*, F_j) \in \mathcal{B}} \sum_i F_{j,i}(\theta_i - \theta_{j,i}^*)^2$
    $\theta \leftarrow \theta - \eta \nabla_\theta (\ell_{\text{NMSE}} + \ell_{\text{EWC}})$
  end for
  Fisher computation for D_k:
  $F_{k,i} \leftarrow \frac{1}{|D_k|} \sum_{(\mathbf{X},\mathbf{H}) \in D_k} (\partial_{\theta_i} \ell_{\text{NMSE}})^2$
  Store snapshot: $\theta_k^* \leftarrow \theta$, $\mathcal{B} \leftarrow \mathcal{B} \cup \{(\theta_k^*, F_k)\}$
end for

guarantee that every measurement—whether collected in the first or the last visited cell—has the same chance of being rehearsed, Algorithm 1 adopts classical *reservoir sampling*:

$$\Pr\big[e_i \in \mathcal{M}_t\big] = \frac{N_{\text{buffer}}}{t}, \quad i = 1, \ldots, t, \tag{6}$$

where $\mathcal{M}_t$ denotes the buffer content after t total observations. The procedure is cell-agnostic and thus well-suited to heterogeneous mobility traces.

1. **Data acquisition.** While the UE resides in scenario k (e.g. UMi-dense), incoming measurement e_t is fed to the reservoir algorithm, which either stores it or discards it with probability $1 - \frac{N_{\text{buffer}}}{t}$.
2. **Mini-batch assembly.** Each stochastic gradient descent (SGD) step draws a batch $\mathcal{B}_{\text{curr}} \subset \mathcal{D}_k$ from the *live* cell trace and a rehearsal batch $\mathcal{B}_{\text{rep}} \subset \mathcal{M}$, then optimizes the mixed NMSE loss of Eq. (4).
3. **UE hand-over.** When the UE moves to the next cell ($k \rightarrow k+1$), the same buffer $\mathcal{M}$ is retained, guaranteeing that legacy UMi statistics remain rehearsed even if those environments are no longer observed.

Loss-Aware Reservoir Sampling (LARS). While uniform reservoir sampling treats all past observations equally, cell-edge channels or deep-fade events (precisely the cases that most hurt throughput) may be under-represented. Algorithm 1 therefore adopts *loss-aware reservoir sampling* (LARS) (Mall et al., 2023; Kumari et al., 2022), which biases the buffer in favor of those channel realizations with which the predictor still struggles. Immediately after performing the forward pass on mini-batch $\mathcal{B}_t$, Algorithm 1 computes the per-observation NMSE $\mathcal{L}_{\text{current}}(\boldsymbol{\theta}, \mathcal{B}_{\text{current}})$ and stores it alongside e_i if the item enters the buffer. When the buffer $\mathcal{M}$ is full ($|\mathcal{M}| = N_{\text{buffer}}$) a newly arrived observation e_t is considered for inclusion with the same reservoir probability $\frac{N_{\text{buffer}}}{t}$. If the decision is *keep*, LARS chooses a *victim* index v according to

$$\Pr[v = i] \;=\; \frac{\big(\ell_i + \epsilon\big)^{-1}}{\sum_{j=1}^{N_{\text{buf}}} (\ell_j + \epsilon)^{-1}}, \tag{7}$$

where $\epsilon > 0$ prevents division by zero. Hence observations whose NMSE has already dropped are *more* likely to be evicted, whereas hard-to-predict channels (e.g. severe multipath or rich scattering) persist longer in $\mathcal{M}$. Retaining difficult samples ensures that the predictor keeps rehearsing rare but performance-critical propagation states (deep fades, high delay spreads, cell-edge SNRs).

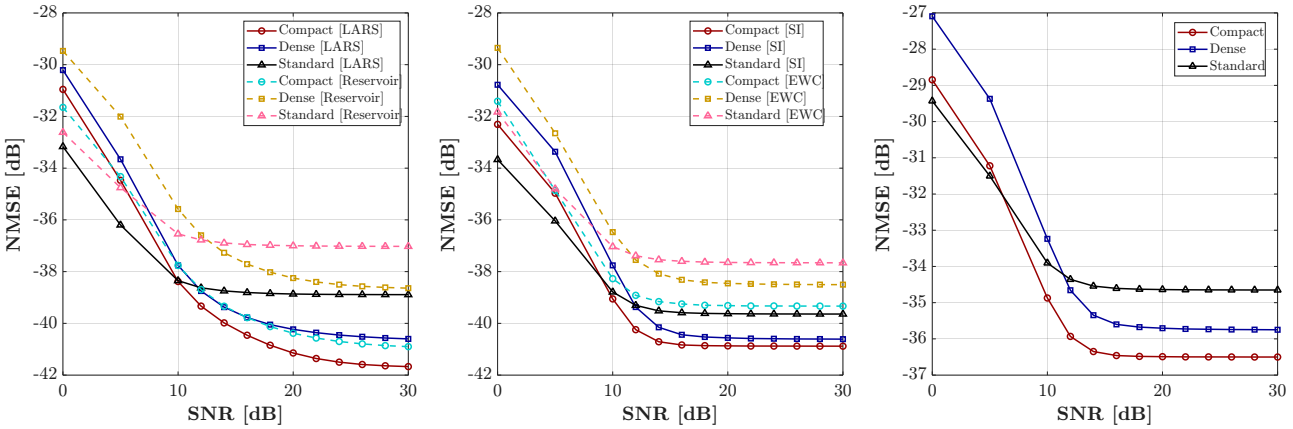
Algorithm 3 Continual Channel Prediction with Learning-without-Forgetting

Require: mixing weight λ, learning rate η, datasets $\{\mathcal{D}_k\}_{k=1}^K$
Initialize model parameters θ
Train on $\mathcal{D}_1$ by minimizing:
$\mathcal{L}_{\text{task}}(\theta) = \frac{1}{|\mathcal{D}_1|} \sum_{(\mathbf{X},\mathbf{H}) \in \mathcal{D}_1} \frac{\|\mathbf{H} - f_\theta(\mathbf{X})\|_F^2}{\|\mathbf{H}\|_F^2}$
$\theta_{\text{old}} \leftarrow \theta$
for $k = 2$ **to** K **do**
  for all minibatch $(\mathbf{X}, \mathbf{H}) \subset \mathcal{D}_k$ **do**
    $\hat{\mathbf{H}} \leftarrow f_\theta(\mathbf{X})$
    $\hat{\mathbf{H}}_{\text{old}} \leftarrow f_{\theta_{\text{old}}}(\mathbf{X})$
    $\mathcal{L}_{\text{task}} \leftarrow \frac{1}{|\mathcal{B}|} \sum \frac{\|\mathbf{H} - \hat{\mathbf{H}}\|_F^2}{\|\mathbf{H}\|_F^2}$
    $\mathcal{L}_{\text{KD}} \leftarrow \frac{1}{|\mathcal{B}|} \sum \frac{\|\hat{\mathbf{H}}_{\text{old}} - \hat{\mathbf{H}}\|_F^2}{\|\hat{\mathbf{H}}_{\text{old}}\|_F^2}$
    $\mathcal{L} \leftarrow \mathcal{L}_{\text{task}} + \lambda \, \mathcal{L}_{\text{KD}}$
    $\theta \leftarrow \theta - \eta \, \nabla_\theta \, \mathcal{L}$
  end for
  $\theta_{\text{old}} \leftarrow \theta$
end for

3.2. Loss Regularization Method

Augmenting the training loss with penalty terms prevents catastrophic forgetting by discouraging large updates to parameters critical for prior tasks (Zhao et al., 2024). We adopt two complementary regularizers: EWC (Yang et al., 2021), which imposes a Fisher-information-weighted quadratic penalty (Calmet & Calmet, 2005), and SI (Zenke et al., 2017), which dynamically accumulates per-weight importance from loss gradients to penalize significant updates, preserving performance across the UMi sequence ({compact, dense, standard}).

Elastic Weight Consolidation (EWC). When the UE goes into a given UMi cell, Algorithm 2 treats the resulting mini-



(a) SNR [dB] vs. NMSE [dB] for experience replay with reservoirs and LARS.

(b) SNR [dB] vs. NMSE [dB] for loss regularization method for SI and EWC.

(c) SNR [dB] vs. NMSE [dB] for LWF.

Figure 3. SNR vs NMSE curves for various continual learning methods tested for cross-network generalization.

dataset $D_k = \{(\mathbf{X}^{(n)}, \mathbf{H}^{(n)})\}_{n=1}^{N_k}$ as one *task*. After training on D_k, Algorithm 2 obtains the weight vector $\boldsymbol{\theta}_k^*$ that minimises the NMSE on that cell's pathloss and fading statistics.

To achieve this, EWC identifies which weights must remain stable by measuring how sensitive the NMSE is to each parameter. Specifically, it estimates the importance of each parameter θ_i for the kth task using an approximation of the diagonal of the Fisher information matrix $F(k, i)$:

$$F_{k,i} = \frac{1}{|D_k|} \sum_{(\mathbf{X},\mathbf{H})\in D_k} \left(\partial_{\theta_i} \mathcal{L}_{\text{NMSE}}(\boldsymbol{\theta}_k^*; \mathbf{X}, \mathbf{H})\right)^2 \quad (8)$$

The expression inside parenthesis, $\partial_{\theta_i} \mathcal{L}_{\text{NMSE}}(\boldsymbol{\theta}_k^*; \mathbf{X}, \mathbf{H})$, is the gradient of the NMSE loss with respect to the parameter θ_i, reflecting how sensitively a small change in θ_i perturbs the characteristics of cell k.

When the UE hands over to the next propagation scenario D_{k+1}, Algorithm 2 introduces a quadratic penalty that keeps $\boldsymbol{\theta}$ near $\boldsymbol{\theta}^k$ proportionally to $F(k, i)$. For a single previous task, the EWC regularization term is

$$\mathcal{L}_{\text{EWC}}(\boldsymbol{\theta}) = \frac{\alpha}{2} \sum_i F_{k,i} \left(\theta_i - \theta^{k,i}\right)^2 \quad (9)$$

where $\alpha > 0$ is a stability coefficient ($\alpha = 0.4$) that balances how strongly EWC penalizes deviation from task k's optimum. After several cell visits the UE has encountered $\mathcal{T}_{1:k} = \{$UMi-*compact*, UMi-*dense*, UMi-*standard*$\}$. Algorithm 2 maintains a *bank* of snapshots $\{(\boldsymbol{\theta}_j^*, F_j)\}_{j=1}^k$.

The penalty therefore generalizes to:

$$\mathcal{L}_{\text{EWC}}(\boldsymbol{\theta}) = \frac{\alpha}{2} \sum_i \sum_{j=1}^{k} F_{j,i} \left(\theta_i - \theta_{j,i}^*\right)^2. \quad (10)$$

Equation (10) encourages the predictor to *retain the parameters that capture the key statistical properties of the channel distribution in each environment*, such as those characterizing the compact (canyon-like), dense (urban block), and standard (wide street) scenarios, while still allowing less-critical weights to adapt to the new cell's Doppler or fading patterns.

In conclusion, after training on task D_k, Algorithm 2 computes Fisher information F_k from NMSE gradients and stores $\boldsymbol{\theta}_k^*$ alongside F_k. Algorithm 2 applies aggregated Fisher penalties to preserve parameters vital for past channel conditions while adapting to new fading dynamics. Since $F_{j,i}$ derives directly from NMSE gradients, it naturally targets weights governing high-energy taps, LOS components, and dominant eigenmode features essential for accurate CQI.

Synaptic Intelligence (SI). SI dynamically tracks parameter importance during training without requiring an explicit Fisher computation. Fisher-based weighting often fails in practice, 1) doubles training time with a full extra dataset pass, 2) exhausts GPU memory storing per-parameter importances, and 3) relies on a quadratic local-curvature approximation that becomes unreliable when mini-batch gradients exhibit high stochasticity (van de Ven, 2025), (Puiu, 2022). To achieve a more robust, continual learning strategy, results therefore compare this Fisher approach against SI. SI avoids all three drawbacks by accumulating importance continuously from the same gradients already used for optimization. These drawbacks are magnified in the UMi sequence, so a second pass would double training time; the diagonal Fisher for ~ 15 M parameters would exceed the 24 GB memory budget of a single A6000 GPU. For the ~ 400 UMi channel states, EWC is untenable: each diagonal Fisher for 15 M parameters ($15 \times 10^6 \times 4$ B ≈ 60 MB) costs an extra epoch,

and storing one per state (400×60 MB ≈ 24 GB) already fills an A6000's 24 GB.

By accumulating parameter importance online from the same gradients used for optimization, incurring only one 32 bit float ($\sim$ 4 bytes) per weight and requiring no extra data sweeps, SI lowers the high-SNR NMSE floor by approximately 0.8–1.4 dB ($\approx$ 10% MSE reduction) relative to EWC, making it readily scalable through the UMi-compact $\rightarrow$ UMi-dense $\rightarrow$ UMi-standard pipeline (Zenke et al., 2017). SI assigns to each parameter θ_i a "synaptic importance" ω_i based on how much change in θ_i contributes to reducing the loss on a task, as shown in Algorithm 4. During minibatch D_k (e.g. the UE dwelling in a specific UMi cell), we track, for every weight θ_i, how much that weight actually *helps* reduces the NMSE on the instantaneous channel pair $(\mathbf{X}, \mathbf{H})$:

$$\tilde{\omega}_i = \tilde{\omega}_i + \left(\nabla_{\theta_i} \mathcal{L}_{\text{NMSE}}\right)^2 \eta \tag{11}$$

where $\nabla_{\theta_i} \mathcal{L}_{NMSE}$ is the gradient of the instantaneous mean-squared error loss with respect to θ_i, which normalizes the Frobenius error by $\|\mathbf{H}\|_F^2$, reflecting the link-adaptation metric used by the gNB scheduler. This accumulation $\tilde{\omega}_i$ intuitively measures the total contribution of θ_i to loss reduction during task D_k.

After completing training on D_k, let $\Delta\theta_i = \theta_i - \theta_i^{(0)}$ be the total change in parameter i during this task. Algorithm 4 then finalizes the importance of θ_i for task k by updating

$$\omega_i \mathrel{+}= \frac{\tilde{\omega}_i}{(\Delta\theta_i)^2 + \xi}, \quad \Delta\theta_i = \theta_i - \theta_i^{(0)}. \tag{12}$$

The above update increases ω_i significantly if a large accumulated gradient (large $\tilde{\omega}_i$) managed to cause only a small net change ($\Delta\theta_i$), indicating that θ_i was repeatedly pulled by the loss (i.e., in the case of deep fade samples) but resisted changing, a sign that θ_i is important for maintaining performance, giving evidence that i governs the core fading statistics. We also set $\theta_i^{(0)} \leftarrow \theta_i$ (forward final weights as the reference for the next task D_{k+1}). All ω_i values persist across cells, so the model remembers which parameters matter for, say, the rich scattering UMi-dense layout even after roaming into a compact canyon.

Given the reference parameters $\theta^{(0)}$ from the start of the current task (which equal the parameters after D_k), the SI loss for task D_{k+1} is formulated as:

$$\mathcal{L}_{\text{SI}}(\boldsymbol{\theta}) \;=\; \frac{\beta}{2} \sum_i \omega_i \left(\theta_i - \theta_i^{(0)}\right)^2, \quad \beta = 0.6. \tag{13}$$

with $\beta > 0$ a weighting hyperparameter (Algorithm 4 uses $\beta = 0.6$) analogous to α in EWC. In wireless channel adaptation, this penalty scales each update to θ_i by its importance ω_i, preventing drift from values learned on prior UMi channel conditions. SI accumulates the product of each parameter update and its instantaneous gradient to compute ω_i online with an $\mathcal{O}(1)$ memory footprint per weight, thereby preferentially regularizing parameters whose sustained gradient magnitudes encode dominant path-loss attenuation and delay-spread dynamics.

Algorithm 4 Continual Channel Prediction with Synaptic Intelligence (SI)

Require: learning rate η, SI weight β, damping ξ
Initialize for all i:
$\theta_i, \omega_i \leftarrow 0, \tilde{\omega}_i \leftarrow 0, \theta_i^{(0)} \leftarrow \theta_i$
for each task D_k (UMi *compact* $\rightarrow$ *dense* $\rightarrow$ *standard*) **do do**
  for each mini-batch $(X, H) \in D_k$ **do do**
    $\hat{H} \leftarrow f_\theta(X)$
    $\ell \leftarrow \ell_{\text{NMSE}}(H, \hat{H})$
    $g_i \leftarrow \nabla_{\theta_i} \ell$
    $\theta_i \leftarrow \theta_i - \eta\, g_i \quad \forall i$
    $\tilde{\omega}_i \mathrel{+}= g_i^2\, \eta \quad \forall i$
  end for
  for each parameter i **do do**
    $\Delta\theta_i \leftarrow \theta_i - \theta_i^{(0)}$
    $\omega_i \mathrel{+}= \frac{\tilde{\omega}_i}{(\Delta\theta_i)^2 + \xi}$ {SI update}
    $\tilde{\omega}_i \leftarrow 0, \theta_i^{(0)} \leftarrow \theta_i$
  end for
end for
Add SI penalty to the loss:
$\mathcal{L} = \ell + \frac{\beta}{2} \sum_i \omega_i (\theta_i - \theta_i^{(0)})^2$

3.3. Learning without Forgetting (LwF)

LwF offers a memory-free continual learning approach by distilling knowledge from a frozen copy of the model trained on past environments. After convergence on environment T_1 with dataset $\mathcal{D}_1$, the optimized parameters $\boldsymbol{\theta}$ are cloned as $\boldsymbol{\theta}_{\text{old}} = \boldsymbol{\theta}$ and kept frozen to serve as a fixed "teacher" for subsequent tasks. For each subsequent environment T_k with a dataset $\mathcal{D}_k$, as the UE hands over into each new UMi layout, whether moving from a compact to a dense or into a standard environment, we train the current model $\boldsymbol{\theta}$ on mini-batches $\mathcal{B}_k \subset \mathcal{D}_k$ to minimize a hybrid loss that combines the NMSE between ground-truth channels and the current model's predictions $\mathcal{L}_{\text{task}}$ with a distillation NMSE $\mathcal{L}_{\text{distill}}$ that measures the discrepancy between the current model's outputs and those of the frozen teacher.

Mini-batch loss terms. Let each sample in $\mathcal{B}_k$ be $(\mathbf{X}_i, \mathbf{H}_i)$, the standard NMSE between the current model's and the frozen teacher's predictions on the same inputs will be

$$\mathcal{L}_{\text{distill}}\big(\boldsymbol{\theta}, \boldsymbol{\theta}_{\text{old}}, \mathcal{B}_k\big) = \frac{1}{|\mathcal{B}_k|} \sum_{i \in \mathcal{B}_k} \frac{\|\hat{\mathbf{H}}_i(\boldsymbol{\theta}_{\text{old}}) - \hat{\mathbf{H}}_i(\boldsymbol{\theta})\|_F^2}{\|\hat{\mathbf{H}}_i(\boldsymbol{\theta}_{\mathbf{old}})\|_F^2}$$

Table 1. **Evaluation loss comparison under dynamic continual learning pipelines (sequence length = 32, replay memory size = 5000) [NMSE Loss in dB]**

Continuous Learning Pipelines	Test: UMi Compact			Test: UMi Dense			Test: UMi Standard		
	Trans.	LSTM	GRU	Trans.	LSTM	GRU	Trans.	LSTM	GRU
Experience Replay [LARS]	-41.824	-41.927	-41.737	-40.851	-40.973	-40.719	-38.910	-39.004	-38.890
Experience Replay [Reservoir]	-41.00	-41.004	-40.900	-38.750	-38.954	-38.700	-37.830	-37.885	-37.790
Loss Regularization [SI]	-41.003	-41.042	-40.850	-40.730	-40.834	-40.530	-39.650	-39.731	-39.530
Loss Regularization [EWC]	-39.220	-39.271	-39.150	-38.730	-38.842	-38.510	-37.610	-37.835	-37.550
Learning Without Forgetting	-35.900	-36.500	-35.800	-35.750	-35.847	-35.560	-34.500	-34.673	-34.420

Combined LwF Objective. Algorithm 3 reuses the mixing weight $\lambda \in [0, 1]$ from Experience Replay to balance fitting new channel data versus preserving past behavior:

$$\mathcal{L}_{\text{LwF}}(\boldsymbol{\theta}) = \lambda\, \mathcal{L}_{\text{task}}(\boldsymbol{\theta}, \mathcal{B}_k) + (1-\lambda)\, \mathcal{L}_{\text{distill}}(\boldsymbol{\theta}, \boldsymbol{\theta}_{\text{old}}, \mathcal{B}_k). \tag{14}$$

The hyperparameter λ balances new-cell adaptation and distillation: increasing λ accelerates fitting to current fading and path-loss, while decreasing λ prioritizes alignment with the frozen teacher to preserve prior UMi-cell characteristics. In this way, the model is encouraged to learn the fresh propagation characteristics of each new UMi scenario, capturing its unique path loss, multipath spread, and Doppler effects while still producing outputs that remain consistent to the behaviors learned in earlier layouts. This mechanism averts catastrophic forgetting across compact, dense, and standard scenarios while rapidly adapting to the UE's evolving fading and path-loss statistics. ER replays the hardest fades through a loss-aware buffer; EWC supplements the NMSE loss with a Fisher-weighted penalty that anchors critical weights; LwF distills from a frozen teacher at zero memory cost. Collectively, these strategies span replay, regularization, and memory-free distillation, yielding complementary paths to handover-robust channel prediction.

4. Results

Experience Replay. Figure 3a compares two memory-based rehearsal schemes—uniform *reservoir* sampling and *loss-aware reservoir sampling* (LARS). Across the full SNR sweep (0-30 dB) LARS consistently dominates the uniform buffer in all three propagation regimes. For the most challenging *dense* case, LARS lowers the high-SNR NMSE floor from -39.7 dB to -40.5 dB, a $\approx 20\%$ reduction in residual error, while the *compact* setting benefits even more, reaching -42.1 dB at 25 dB SNR. Because the buffer is biased towards hard-to-predict fades, the predictor rehearses precisely those outliers that dominate the tail of the loss distribution, yielding steeper convergence and a $1-2$ dB gap over the uniform baseline throughout the mid-SNR region. These results confirm that targeted replay is crucial for retaining accurate channel dynamics when the user equipment (UE) traverses heterogeneous cell configurations.

Loss Regularization Method. Figure 3b evaluates Fisher-based EW against SI. While both mechanisms suppress catastrophic drift, SI achieves a uniformly lower NMSE, e.g. -40.8 dB versus -39.3 dB in the *compact* layout at 20dB SNR. SI's online importance tracking penalizes updates to weights with high past importance, enabling adaptation of peripheral parameters while preserving core ones. As a consequence, SI preserves the sharp error drop observed around 8-12dB without the saturation seen for EWC, yielding an additional $0.8-1.4$ dB gain at high SNR and a 10% mean-squared-error reduction across the three tasks.

Learning Without Forgetting. Figure 3c shows that *LWF* delivers the *highest* residual NMSE of the three continual-learning methods—e.g., -36.9 dB (*compact*), -35.8 dB (*dense*) and -34.4 dB (*standard*) at 25dB SNR—lagging LARS and SI by roughly $0.7-1.5$ dB. Thus, even though its distillation-only strategy is less effective than replay or weight anchoring, LwF provides a lightweight improvement over no adaptation at all, requiring neither memory buffers nor second-order statistics and remaining attractive for resource-constrained deployments.

5. Conclusion

Our study demonstrates that continual learning substantially improves cross-cell channel prediction: loss-aware experience replay and synaptic intelligence lower the high-SNR NMSE floor by up to 2dB ($\approx$3%) relative to naïve fine-tuning, while even memory-free LwF yields a consistent 1dB gain as shown in Table 1. These results confirm that rehearsing loss-critical fades and selectively anchoring influential weights are key to retaining past knowledge without sacrificing plasticity. The proposed framework therefore offers a practical path toward hand-over-robust CSI prediction, easing deployment across heterogeneous network topologies. For additional hyperparameter sensitivity studies, including sequence length effects and buffer-size ablations, see Appendix C (Tables 3 and 4). Future work will focus on integrating the replay schedulers into 3GPP NR Channel State Information-Reference Signal (CSI-RS) procedures, quantifying fronthaul and UE memory overhead, and standardizing task-aware buffers within the O-RAN RIC for

online adaptation. The approach scales to multi-cell joint prediction and can be benchmarked on upcoming 3GPP Release-19 indoor-hotspot and sub-6 GHz/sub-THz channel models, advancing standard-compliant, continually learning base-station intelligence.

References

Bhattacharya, S., Mohsin, M. A., Rajabalifardi, K., and Cioffi, J. M. Successive interference cancellation-aided diffusion models for joint channel estimation and data detection in low rank channel scenarios, 2025. URL https://arxiv.org/abs/2501.11229.

Calmet, X. and Calmet, J. Dynamics of the fisher information metric. *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics*, 71(5):056109, 2005.

Dey, R. and Salem, F. M. Gate-variants of gated recurrent unit (gru) neural networks. In *2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS)*, pp. 1597–1600. IEEE, 2017.

Fedus, W., Ramachandran, P., Agarwal, R., Bengio, Y., Larochelle, H., Rowland, M., and Dabney, W. Revisiting fundamentals of experience replay. In *International conference on machine learning*, pp. 3061–3071. PMLR, 2020.

Fujimoto, S., Meger, D., and Precup, D. An equivalence between loss functions and non-uniform sampling in experience replay. *Advances in neural information processing systems*, 33:14219–14230, 2020.

Garcia-Saavedra, A. and Costa-Perez, X. O-ran: Disrupting the virtualized ran ecosystem. *IEEE Communications Standards Magazine*, 5(4):96–103, 2021.

Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., and Schmidhuber, J. Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28 (10):2222–2232, 2016.

Han, K., Xiao, A., Wu, E., Guo, J., Xu, C., and Wang, Y. Transformer in transformer. *Advances in neural information processing systems*, 34:15908–15919, 2021.

Jaeckel, S., Raschkowski, L., Börner, K., and Thiele, L. Quadriga: A 3-d multi-cell channel model with time evolution for enabling virtual field trials. *IEEE transactions on antennas and propagation*, 62(6):3242–3256, 2014.

Jiang, H., Cui, M., Ng, D. W. K., and Dai, L. Accurate channel prediction based on transformer: Making mobility negligible. *IEEE Journal on Selected Areas in Communications*, 40(9):2717–2732, 2022.

Jiang, W. and Schotten, H. D. Neural network-based fading channel prediction: A comprehensive overview. *IEEE Access*, 7:118112–118124, 2019.

Jiang, W. and Schotten, H. D. Recurrent neural networks with long short-term memory for fading channel prediction. In *2020 IEEE 91st vehicular technology conference (VTC2020-Spring)*, pp. 1–5. IEEE, 2020.

Joo, J., Park, M. C., Han, D. S., and Pejovic, V. Deep learning-based channel prediction in realistic vehicular communications. *IEEE Access*, 7:27846–27858, 2019.

Kemker, R., McClure, M., Abitino, A., Hayes, T., and Kanan, C. Measuring catastrophic forgetting in neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Kim, C. D., Jeong, J., and Kim, G. Imbalanced continual learning with partitioning reservoir sampling. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIII 16*, pp. 411–428. Springer, 2020.

Kim, H., Choi, J., and Love, D. J. Machine learning for future wireless communications: Channel prediction perspectives. *arXiv preprint arXiv:2502.18196*, 2025.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Krutsylo, A. Batch sampling for experience replay. In *Proceedings of the 7th Joint International Conference on Data Science & Management of Data (11th ACM IKDD CODS and 29th COMAD)*, pp. 202–206, 2024.

Kumari, L., Wang, S., Zhou, T., and Bilmes, J. A. Retrospective adversarial replay for continual learning. *Advances in neural information processing systems*, 35:28530–28544, 2022.

Lee, T., Park, J., Kim, H., and Andrews, J. G. Generating high dimensional user-specific wireless channels using diffusion models, 2024. URL https://arxiv.org/abs/2409.03924.

Li, H., Ding, L., Wang, Y., Wu, P., and Wang, Z. Impact of channel aging on massive mimo vehicular networks in non-isotropic scattering scenarios. In *2021 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6. IEEE, 2021.

Li, Z. and Hoiem, D. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

Liu, B., Liu, X., Gao, S., Cheng, X., and Yang, L. Llm4cp: Adapting large language models for channel prediction. *Journal of Communications and Information Networks*, 9 (2):113–125, 2024.

Liu, G., Xu, Y., Zongjiang, H., Rao, Y., Xia, J., and Fan, L. Deep learning-based channel prediction for edge computing networks toward intelligent connected vehicles. *IEEE Access*, PP:1–1, 08 2019. doi: 10.1109/ACCESS.2019.2935463.

Liu, Q., Wang, P., Sun, J., Li, R., and Li, Y. Wireless channel prediction of gru based on experience replay and snake optimizer. *Sensors*, 23(14):6270, 2023.

Maei, H. R., Szepesvári, C., Bhatnagar, S., and Sutton, R. S. Toward off-policy learning control with function approximation. In *ICML*, volume 10, pp. 719–726, 2010.

Mall, U., Hariharan, B., and Bala, K. Change-aware sampling and contrastive learning for satellite images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5261–5270, 2023.

Polese, M., Giordani, M., and Zorzi, M. 3gpp nr: the standard for 5g cellular networks. *5G Italy White eBook: from Research to Market*, 2018.

Puiu, C. O. Rethinking exponential averaging of the fisher. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 327–343. Springer, 2022.

Rolnick, D., Ahuja, A., Schwarz, J., Lillicrap, T., and Wayne, G. Experience replay for continual learning. *Advances in neural information processing systems*, 32, 2019.

van de Ven, G. M. On the computation of the fisher information in continual learning. *arXiv preprint arXiv:2502.11756*, 2025.

Villena-Rodriguez, A., Martín-Vega, F. J., Gómez, G., Aguayo-Torres, M. C., and Kaddoum, G. Aging-resistant wideband precoding in 5g and beyond using 3d convolutional neural networks. *arXiv preprint arXiv:2407.07434*, 2024.

Yang, X., Li, F., Li, T., Ji, W., and Liang, Y. Elastic weight consolidation continual learning based signal detection in multiple channel mimo system. In *2021 IEEE/CIC International Conference on Communications in China (ICCC)*, pp. 534–539. IEEE, 2021.

Zenke, F., Poole, B., and Ganguli, S. Continual learning through synaptic intelligence. In *International conference on machine learning*, pp. 3987–3995. PMLR, 2017.

Zhang, S., Zhang, S., Mao, Y., Yeung, L. K., Clerckx, B., and Quek, T. Q. Transformer-based channel prediction for rate-splitting multiple access-enabled vehicle-to-everything communication. *IEEE Transactions on Wireless Communications*, 2024.

Zhao, X., Wang, H., Huang, W., and Lin, W. A statistical theory of regularization-based continual learning. *arXiv preprint arXiv:2406.06213*, 2024.

Zhou, Y., Tian, X., Zhang, C., Zhao, Y., and Li, T. Elastic weight consolidation-based adaptive neural networks for dynamic building energy load prediction modeling. *Energy and Buildings*, 265:112098, 2022.

Zilberstein, N., Swami, A., and Segarra, S. Joint channel estimation and data detection in massive mimo systems based on diffusion models. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 13291–13295. IEEE, 2024.

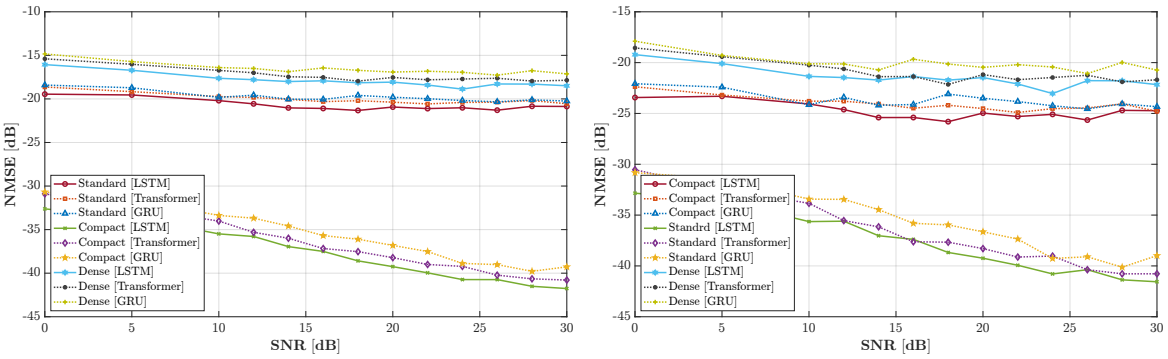
A. Baseline Distributions and Results

First, discussion on the baseline models' architectures as discussed in Section 2.2. Finding here elaborates on the architectures for the GRU and the Transformer and then discusses their performance in Figure 4's baseline scenarios.

GRU. The GRU variant retains the same preprocessing pipeline as the LSTM—each T-length window $\mathbf{X} \in \mathbb{C}^{2\times T\times N_{\mathrm{tx}}\times N_{\mathrm{rb}}\times N_{\mathrm{rx}}}$ is reshaped into a length-$d_{\text{in}} = 2N_{\mathrm{tx}}N_{\mathrm{rb}}N_{\mathrm{rx}}$ feature vector per time step—but the temporal backbone is a three-layer Gated Recurrent Unit with hidden width $d_{\text{hid}} = 32$. Compared with the LSTM, the GRU merges the input and forget gates, cutting the recurrent parameter count roughly by one-third while still maintaining gating dynamics that model slow and fast fading jointly. After processing, the final hidden state $\mathbf{h}_T \in \mathbb{R}^{d_{\text{hid}}}$ is mapped through a linear layer of size $(2N_{\mathrm{tx}}N_{\mathrm{rb}}N_{\mathrm{rx}}) \times d_{\text{hid}}$ and reshaped back to $\hat{\mathbf{H}} \in \mathbb{C}^{2\times N_{\mathrm{tx}}\times N_{\mathrm{rb}}\times N_{\mathrm{rx}}}$. This leaner gating structure delivers competitive NMSE with shorter inference latency, making it attractive for edge deployment when computational budgets are tight.

Transformer. For longer temporal horizons, evaluation includes a lightweight Transformer that first flattens each spatial slice into a $d_{\text{in}} = 2N_{\mathrm{tx}}N_{\mathrm{rb}}N_{\mathrm{rx}}$ vector, projects it to a $d_{\text{model}} = 128$ embedding, and enriches it with a multi-frequency positional encoding tailored to wireless spectra. A single encoder layer and a single decoder layer, each with four self-attention heads, form the core sequence-to-sequence module; a learned start token serves as the one-step decoder query. The decoder output is passed through a final linear layer back to $2N_{\mathrm{tx}}N_{\mathrm{rb}}N_{\mathrm{rx}}$ and reshaped to the predicted channel matrix $\hat{\mathbf{H}}$. The self-attention mechanism allows the model to capture dependencies across the entire T-slot context without recurrence, enabling highly parallel training; however, its larger projection matrices and quadratic attention cost demand more memory than the recurrent baselines, so results limit depth to one layer each to stay within gNB resource constraints while still harnessing the Transformer's global receptive field.

LSTM. A 3-layer LSTM network takes as input a tensor $\mathbf{X} \in \mathbb{C}^{2\times T\times N_{\mathrm{tx}}\times N_{\mathrm{rb}}\times N_{\mathrm{rx}}}$, where the first dimension separates real and imaginary parts and T is the look-back window. At each time step, the spatial slice $\big(2, N_{\mathrm{tx}}, N_{\mathrm{rb}}, N_{\mathrm{rx}}\big)$ flattens to a vector of length $2N_{\mathrm{tx}}N_{\mathrm{rb}}N_{\mathrm{rx}}$, yielding a sequence whose feature size we denote d_{in}. This sequence is processed by a stack of $n_{\text{layers}} = 3$ LSTM blocks with hidden width $d_{\text{hid}} = 32$, producing hidden states $\mathbf{h}_t \in \mathbb{R}^{d_{\text{hid}}}$ for $t = 1, \ldots, T$. The final state $\mathbf{h}_T$ is passed through a fully connected layer $W \in \mathbb{R}^{(2N_{\mathrm{tx}}N_{\mathrm{rb}}N_{\mathrm{rx}})\times d_{\text{hid}}}$ to predict the next-slot channel matrix $\hat{\mathbf{H}} \in \mathbb{C}^{2\times N_{\mathrm{tx}}\times N_{\mathrm{rb}}\times N_{\mathrm{rx}}}$, after which the original tensor shape is restored. The modest hidden width keeps the parameter count low—about $4d_{\text{hid}}\big(d_{\text{in}} + d_{\text{hid}}\big)$ per layer—so the model fits comfortably in gNB memory while still capturing the essential channel dynamics. LSTMs excel in this setting because wireless channels form a strongly time-correlated sequence governed by user mobility and multipath evolution; the forget, input, and output gates allow the network to preserve long-range dependencies (slow fading) while rapidly adapting to short-term variations (fast fading) within the same recurrent architecture.



(a) Baseline SNR trained on UMi compact and tested on all scenarios under all architectures.

(b) Baseline SNR trained on UMi standard and tested on all scenarios under all architectures.

Figure 4. Prediction error under baseline conditions when tested under zero shot data settings.

Table 2. Configuration Parameters for 3GPP Urban Microcell (UMi) and Urban Macrocell (UMa) Scenarios

(a) UMi

Parameter	Standard	Dense	Compact
Carrier Frequency	5 GHz	5 GHz	5 GHz
Bandwidth	100 MHz	100 MHz	100 MHz
Antenna Tilt (°)	30	10	0
Element Spacing (λ)	0.50	0.25	1.00
Rx Antenna Type	dipole	patch	cross_pol
Rx Polarization	$\pm45°$	H/V	$\pm45°$
Distance Range (m)	[50, 100]	[20, 60]	[120, 200]
Tx Height (m)	10	6	15
UE Height (m)	1.5	1.0	2.0

(b) UMa

Parameter	Standard	Large–H/V	Small–V
Carrier Frequency	2.6 GHz	2.6 GHz	2.6 GHz
Antenna Tilt (°)	12	10	15
Element Spacing (λ)	0.50	0.60	0.50
Tx Array Size (M×N)	8×4	10×6	6×2
Rx Antenna Type	dipole	patch	omni
Rx Polarization	$\pm45°$	H/V	V
Distance Range (m)	[100, 500]	[100, 500]	[100, 500]
Tx Height (m)	25	25	25
UE Height (m)	1.5	1.5	1.5

B. Network Distributions.

B.1. Urban Macro Channel [UMa]

For the UMa data set is synthesised with QuaDRiGa at $f_c = 2.6$ GHz and 20 MHz bandwidth, emulating macro base stations mounted 25 m above street level that illuminate users at radial distances between 100 and 500 m. Three antenna-array configurations—*standard* (8×4 panel, dual-pol dipoles), *large–H/V* (10×6 panel, quad-pol patches), and *small–V* (6×2 compact omni array)—capture a range of sector-capacity trade-offs, as summarised in Table 2. Each Monte-Carlo realisation deploys 256 UEs whose linear tracks are sampled at 30 time instants and 18 OFDM resource blocks, yielding tensors of dimension $[30 \times N_{\text{tx}} \times 18 \times N_{\text{rx}} \times 256]$. Although the simulation seed is fixed to ensure reproducibility, every iteration perturbs the UE starting positions, and the LOS/NLOS state is drawn from the 3GPP distance-dependent law $\Pr_{\text{LOS}}(d) = e^{-d/300}$. This controlled displacement makes successive channel snapshots partially correlated, challenging the prediction network to learn both slow macro-scale trends and fast small-scale fading across the diverse UMa configurations.

B.2. Urban Micro Channel [UMi]

The UMi data set used throughout the paper is generated with the QuaDRiGa Monte-Carlo engine configured for 100 MHz bandwidth at $f_c = 5$ GHz and an 8×2 MIMO link (eight dual-polarised transmit elements arranged as a 2×2 panel and a two-element UE array). Three propagation "flavors" are synthesized: *standard*, *dense*, and *compact* as shown in Figure 5, which differ only in antenna downtilt, inter-element spacing, handset height, and the underlying 3GPP/5G channel profile (LOS or NLOS); the complete parameter list is given in Table 2. Each simulation realizes 256 user equipment whose linear tracks are discretized into 500 time instants and 18 OFDM resource blocks, producing a complex-valued tensor of size $\big[500 \times 2 \times 18 \times 8 \times 256\big]$. Although a fixed random seed guarantees repeatability, we introduce a small random displacement to every UE position at each Monte-Carlo iteration; this causes consecutive tensors to share local scatterers and therefore exhibit pronounced spatial-temporal correlation. The channel-prediction model must learn these correlations to extrapolate reliably from the recent T-slot history to the next-slot channel matrix, especially when the UE migrates between the three UMi scenarios.

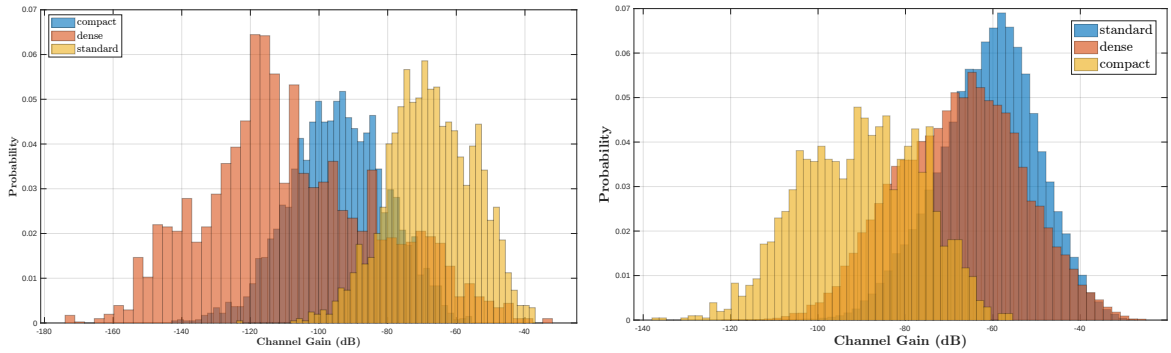
B.3. Dynamic Parametric Changes

1. Correlation among time series sequence. For each UE the script draws a random radius $d \sim \mathcal{U}[d_{\min}, d_{\max}]$ and azimuth $\phi \sim \mathcal{U}[0, 2\pi)$, fixes the initial position

$$\mathbf{p}_0 = \big[d\cos\phi,\ d\sin\phi,\ h_{\text{ue}}\big]^\top,$$

and then assigns the linear QuaDRiGa track `qd_track('linear',L,phi)` with length $L = 2\,\text{m}$. The call `tr.interpolate_positions(num_snap-1,'time')` creates equally spaced snapshots

$$\mathbf{p}_t = \mathbf{p}_0 + \frac{t}{T-1}\,L\,[\cos\phi,\ \sin\phi,\ 0]^\top, \quad t = 0, \ldots, T-1,\ T = 500,$$



(a) Probability distribution for UMi with paramters described in table 2(a)

(b) Probability distribution for UMa with paramters described in table 2(b).

Figure 5. Probability distribution deviation under different network configurations.

so the spatial step is $\Delta s = L/(T-1) \approx 4\,\text{mm}$ $(= \sim\lambda/15$ at $f_c = 5\,\text{GHz})$. Under the WSSUS assumption an isotropic ring of scatterers of radius R_{sc} produces the small-scale correlation as shown in Figure 6.

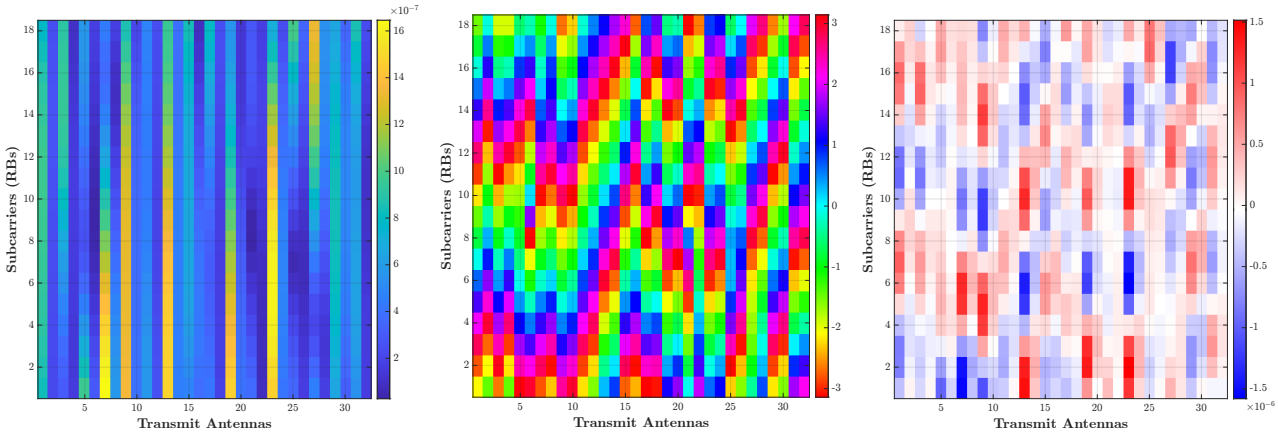
$$\rho_h(\Delta\mathbf{r}) = \mathbb{E}\big[h(\mathbf{r})\, h^*(\mathbf{r}+\Delta\mathbf{r})\big] = J_0\Big(\frac{2\pi}{\lambda}\,\|\Delta\mathbf{r}\|\Big),$$

so between successive snapshots $\rho_h(\Delta s) \approx J_0(2\pi/15) \gtrsim 0.97$—*very* strong. Because the UE displacement is deterministic, the channel at time t

$$H_t(k,m) = \sum_{\ell=1}^{L_{\text{cl}}} \alpha_\ell e^{-j2\pi k\Delta f\,\tau_\ell} e^{+j\frac{2\pi}{\lambda} d_t m \sin\theta_{\ell,t}} e^{-j\frac{2\pi}{\lambda}\mathbf{k}_\ell \mathbf{p}_t}$$

inherits this coherence: the last term adds a linear phase $\propto (\mathbf{k}_\ell \cdot \mathbf{v})\, t$ whose slope is so gentle that adjacent Monte-Carlo realisations remain *temporally self-correlated*. Consequently

- the magnitude heat-map of Figure 6a shows nearly vertical stripes—frequency coherence persists because $\tau_{\ell,t}$ changes negligibly over Δs;
- the phase map Figure 6b exhibits diagonal ramps whose gradient equals the deterministic spatial phase drift $\frac{2\pi}{\lambda} d_t \sin\theta_{\ell,t}$ multiplied by the constant step Δs;

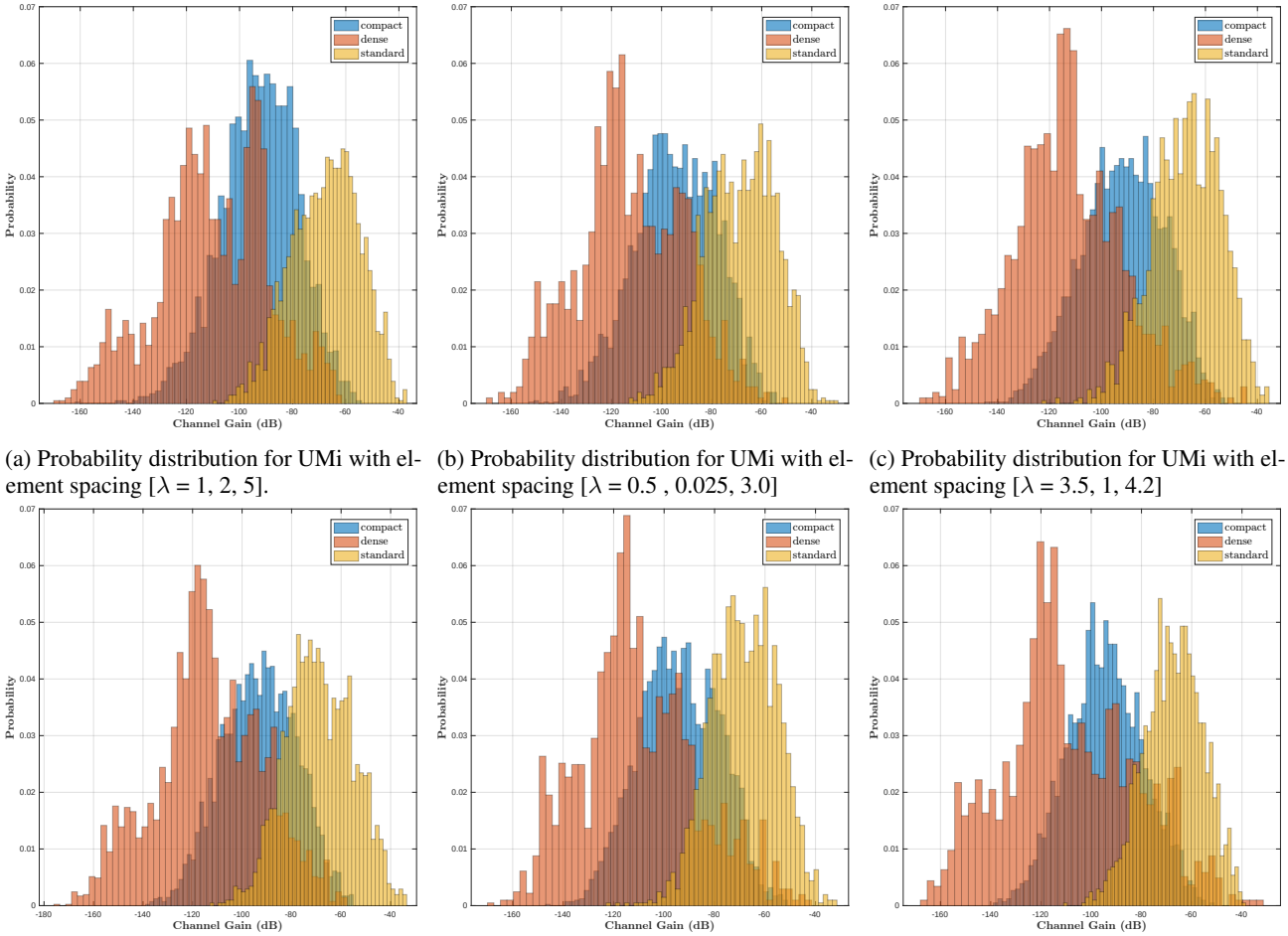


(a) Transmit antenna vs Subcarriers for magntide of the complex channel matrix **H**.

(b) Transmit antenna vs Subcarriers for phase of the complex channel matrix **H**.

(c) Transmit antenna vs Subcarriers for real part of the complex channel matrix **H**.

Figure 6. Correlation among transmit anetnna vs. subcarriers for user 1 and time stamp 1.



(a) Probability distribution for UMi with element spacing [λ = 1, 2, 5].

(b) Probability distribution for UMi with element spacing [λ = 0.5 , 0.025, 3.0]

(c) Probability distribution for UMi with element spacing [λ = 3.5, 1, 4.2]

(d) Probability distribution for UMi with antenna tilt [° = 0, 90, 120].

(e) Probability distribution for UMi with antenna tilt [° = 0, 10 , 15]

(f) Probability distribution for UMi with antenna tile [° = 0, 100, 200]

Figure 7. Effect of element spacing and antenna tilt on probability distribution of channel gains under UMi scenario.

- the real part in Figure 6c reveals blocks of coherent (additive) or destructive interference depending on whether the phase difference between neighbouring antennas is near 0 or π.

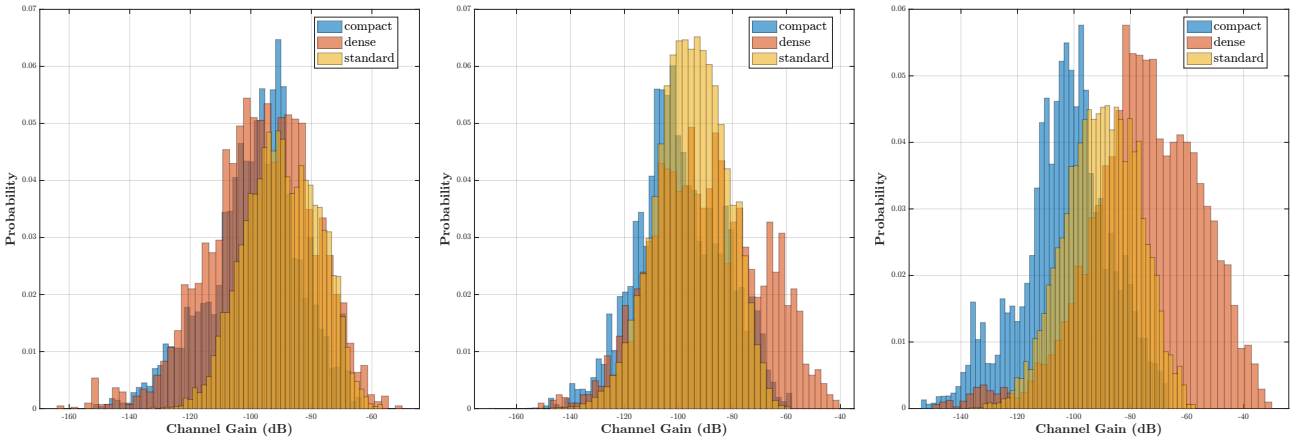
In sum, the *incremental UE motion* $\mathbf{p}_{t+1} - \mathbf{p}_t = \Delta s\,[\cos\phi, \sin\phi, 0]^\top$ couples the spatial, frequency and *time* dimensions: adjacent snapshots share scatterers and remain highly correlated, providing the predictive model with exploitable temporal structure.

2. Effect of Element Spacing. Increasing the inter-element spacing $d\lambda$ widens the physical aperture of the array—yielding an L/λ-dependent array gain that shifts the entire channel gain histogram to the right—while simultaneously driving down the spatial correlation terms $\rho_k(d) = J_0(2\pi dk)$ in the transmit/receive correlation matrix. Because the variance of the instantaneous power $g = \|\mathbf{R}^{1/2}(d)\mathbf{w}\|^2$ is $2\sum_k (N-k)\rho_k^2(d)$, weaker correlations at larger d shrink this variance, producing a tighter distribution with fewer deep fades. Hence, "dense" arrays with $d \lesssim 0.5\lambda$ exhibit the left-shifted, broad brown histograms observed, "compact" ($d \approx 1\lambda$) fall in the middle, and "standard" spacings ($d \geq 2\lambda$) give the right-shifted, sharply peaked yellow curves as shown in Figure 7.

3. Effect of Antenna Tilt. Downtilting the base-station panel multiplies every small-scale channel coefficient by the vertical antenna pattern

$$G(\theta) = G_{\max} - \min\Big\{\, 12\big(\tfrac{\theta-\theta_{\text{tilt}}}{\theta_{3\text{dB}}}\big)^2,\ \text{SLA}_V \Big\}\ [\text{dB}],$$

with half-power beam-width $\theta_{3\text{dB}} \approx 10^\circ$ and a side-lobe cap $\text{SLA}_V \approx 20$ dB. For a UE at horizontal range R and height h_{ue}



(a) Channel gain vs. Probability distribution for scenario A.

(b) Channel gain vs. Probability distribution for scenario B.

(c) Channel gain vs. Probability distribution for scenario C.

Figure 8. Combined effect of cross network configurtion on channel distributions.

below the array (h_{tx}), the elevation angle is $\theta(R) = \tan^{-1}\big[(h_{\text{tx}} - h_{\text{ue}})/R\big]$, so the large-scale channel gain

$$g_{\text{LS}}(R) = G\big(\theta(R)\big) - 10\,\alpha \log_{10} R + X_\sigma \quad [\text{dB}],$$

combines tilt-dependent antenna gain, distance-dependent path-loss ($\alpha \approx 3.1$ in UMi NLOS) and log-normal shadowing $X_\sigma \sim \mathcal{N}(0, \sigma^2)$. With UEs roughly uniform in range $[R_{\min}, R_{\max}]$, the elevation pdf is

$$f_\Theta(\theta) = \frac{(h_{\text{tx}} - h_{\text{ue}}) \cos^2\theta}{(R_{\max} - R_{\min})(h_{\text{tx}} - h_{\text{ue}}) \cos\theta + R_{\min}},$$

so the variance $\text{Var}\big[G(\Theta)\big] = \mathbb{E}[G^2(\Theta)] - \mathbb{E}^2[G(\Theta)]$ decreases as θ_{tilt} increases because the effective support of Θ shrinks to the nearly flat main-lobe. Hence large downtilt ($\sim 30°$) aligns the main lobe with nearby UEs, producing higher median gains and a tighter histogram (yellow "standard" curves); little or no downtilt ($0-10°$) illuminates farther ranges, yielding lower, more spread-out gains (brown "dense" curves); intermediate tilts (blue "compact") lie between these extremes. The curves are shown in Figure 7.

4. Combined Effect of Cross Parameterization. Experiments examine three representative environments whose antenna deployments differ widely in aperture, element spacing, height, downtilt, and user range.

Scenario A. This baseline group follows the canonical 3GPP-UMi geometry. The *standard_A* cell employs an 8×2 dual-polarised panel ($MN = 16$ elements) mounted at $h_{\text{tx}} = 35$ m, downtilted by $6°$ and spaced at $d_t = \frac{2}{3}\lambda$; UEs move at ranges $R \in [80, 150]$ m under the LOS profile *3GPP_38.901_UMi_LOS*. The *dense_A* flavour shrinks the array to a 4×1 ($MN = 4$) patch with no downtilt and very tight 0.25λ spacing, placed at $h_{\text{tx}} = 10$ m; pedestrians roam only $R \in [10, 60]$ m in NLOS (*3GPP_38.901_UMi_NLOS*). Finally, the *compact_A* cell restores an 8×4 panel ($MN = 32$) but widens the spacing to 2.5λ and downtilts by $30°$; at $h_{\text{tx}} = 25$ m it serves users at $R \in [40, 100]$ m with the LOS-rich *5G-ALLSTAR_DenseUrban_LOS* profile.

Scenario B. This setting exaggerates macro/ hot-spot contrast. The *standard_B* macro site is a 16×4 array ($MN = 64$) on a 45 m rooftop, downtilt $8°$, wide 1.2λ spacing, covering $R \in [150, 300]$ m in LOS. Opposite to it, the *dense_B* pedestrian hot-spot uses just 2×1 elements ($MN = 2$) at street-lamp height $h_{\text{tx}} = 6$ m, zero tilt, ultra-tight 0.15λ spacing and NLOS users at $R \in [5, 30]$ m. Mid-way, the *compact_B* cell keeps an 8×4 panel ($MN = 32$) with *huge* 3λ spacing and strong $35°$ downtilt, serving $R \in [40, 120]$ m under LOS.

Scenario C. A suburban/indoor mix spanning indoor hotspots to rooftop cells: *standard_C*: 8×2 array ($MN = 16$) at $h_{\text{tx}} = 25$ m, no downtilt, spacing 1λ, LOS users $R \in [100, 180]$ m. *dense_C*: indoor 4×2 panel ($MN = 8$) at $h_{\text{tx}} = 3$ m, upward tilt $-15°$, spacing 0.5λ, NLOS users $R \in [2, 15]$ m. *compact_C*: rooftop 12×3 array ($MN = 36$) at $h_{\text{tx}} = 30$ m, downtilt $20°$, spacing 2λ, LOS users $R \in [60, 140]$ m.

The wide variation in array aperture, element spacing, antenna height, downtilt angle, and user-to-base-station distance across Scenarios A-C produces fundamentally different channel "shapes" in each environment as shown in Figure 8. For instance, a dense urban hotspot (e.g., *dense_C*) uses tightly spaced indoor antennas and very short links, resulting in richly scattered, rapidly varying multipath that is hard to predict. By contrast, a rooftop macrocell (e.g., *standard_C*) with wider spacing, higher mounting, and longer ranges yields smoother, largely line-of-sight channels. Steep downtilts and greater heights enhance broad coverage ideal in suburban settings, while close spacing and low mounts capture fine multipath detail critical for indoor NLOS operation. Since our neural predictor internalizes the specific gain distribution, delay spread, and angular statistics of one scenario, it struggles when faced with a channel whose clutter density, pathloss characteristics, and angular spread differ significantly. In simple terms, changing physical parameters reshapes how signals bounce and fade; without exposure to each unique "network configuration settings," model accuracy (NMSE) and temporal forecasting degrade significantly.

C. Hyperparameter Sensitivity in Continual Learning

C.1. Impact of Sequence Length on Continual Learning Pipelines

Table 3 compares the NMSE for the above-discussed five dynamic continual-learning pipelines with sequence length=16. In every case, the LSTM backbone outperforms the Transformer, which in turn outperforms the GRU. Performance degrades smoothly from the Compact to Dense to Standard environments. Moreover, both ER variants (LARS and Reservoir) consistently achieve the lowest NMSE, followed by EWC, with lwf exhibiting the highest errors.

Table 3. **Evaluation loss comparison under dynamic continual learning pipelines (sequence length = 16, replay memory size = 5000) [NMSE Loss in dB]**

Continuous Learning Pipelines	Test: UMi Compact			Test: UMi Dense			Test: UMi Standard		
	Trans.	LSTM	GRU	Trans.	LSTM	GRU	Trans.	LSTM	GRU
Experience Replay [LARS]	-41.624	-41.727	-41.537	-40.651	-40.773	-40.519	-38.710	-38.804	-38.690
Experience Replay [Reservoir]	-40.800	-40.804	-40.700	-38.550	-38.754	-38.500	-37.630	-37.685	-37.590
Loss Regularization [SI]	-40.803	-40.842	-40.650	-40.530	-40.634	-40.330	-39.450	-39.531	-39.330
Loss Regularization [EWC]	-39.020	-39.071	-38.950	-38.530	-38.642	-38.310	-37.410	-37.635	-37.350
Learning Without Forgetting	-35.700	-36.300	-35.600	-35.550	-35.647	-35.360	-34.300	-34.473	-34.220

C.2. Effect of Reduced Replay Buffer Size on ER Performance

Table 4 reports the NMSE for ER with the memory buffer reduced from 5000 to 3000 and sequence $length = 32$. As before, the LSTM backbone consistently outperforms the Transformer, which in turn beats the GRU, and performance degrades smoothly from the Compact to Dense to Standard UMi scenarios.

Table 4. **Evaluation loss for Experience Replay pipelines with reduced memory ($M = 3000$) and sequence length = 32 [NMSE Loss in dB]**

ER Variant	Test: UMi Compact			Test: UMi Dense			Test: UMi Standard		
	Trans.	LSTM	GRU	Trans.	LSTM	GRU	Trans.	LSTM	GRU
Experience Replay [LARS]	-40.317	-41.289	-39.137	-39.682	-40.527	-38.314	-38.201	-38.864	-36.873
Experience Replay [Reservoir]	-39.612	-40.574	-38.432	-37.976	-38.815	-36.752	-36.443	-37.311	-35.216