Machine learning design for wireless communication

Arash Behboodi
Engineer, Sr Staff/Mgr
Qualcomm Technologies Netherlands B.V.
Qualcomm AI Research
AI for wireless is here

Qualcomm 5G AI Suite
next-generation 5G performance enhancements

Qualcomm prototypes AI-enabled Air Interface

More showcases in MWCs

Snapdragon is a product of Qualcomm Technologies, Inc. and/or its subsidiaries.
Machine Learning vs Wireless

Accurate prediction: design with real world priors, fast and flexible models

Accurate modeling: learning generative process

Inverse models: learned optimizers

sensing and perception

Information theory as mathematical framework

Tractable mathematical models e.g., Gaussian, Rayleigh

signal processing statistics optimization

Interpretable solutions

Good generalization under different deployment condition

Simple model adaptation (e.g. different antenna, pilot patterns, SNR, Doppler and delay spread)
Machine learning for wireless communication: challenges

Out-of-domain generalization
The models trained for a specific task should generalize well or adapt to new unseen scenarios
Ex. Unseen dopplers, channel condition

Supervised learning
Supervised learning is costly and at times infeasible
Ex: fingerprinting localization in dynamic environments

Adaptive ML models
ML models should adapt to different scenarios
Ex: different antenna configurations, channel condition

Other issues: interpretability, causality, efficient learning, theoretical guarantees
Neural Augmentation

Boosting domain expertise with data-driven knowledge

**Machine Learning**
- **Benefits:** Modeling flexibility, computational efficiency
- **Challenges:** out-of-domain generalization, interpretability

**Domain Knowledge**
- **Benefits:** Out-of-domain generalization, interpretability
- **Challenges:** Complex modeling, processing complexity

**Data versus inductive bias**
Inductive bias helps generalization.

**Generative versus inverse modeling**
Generative models for data generation process
Inverse modeling to infer model parameters

- Convolutions in computer vision
- Transformers in NLP
- Simulators
- Graphical models
A small detour: equivariance

Integrating inductive bias from data symmetry to the model

• Convolutional kernels are designed based on translation equivariance - Convolutional neural networks are built based on translation invariance of the classification task

Equivariance property: A function \( f: V_{in} \rightarrow V_{out} \) is equivariant w.r.t to group \( G \), if

\[
f(\rho_g^{in} \cdot x) = \rho_g^{out} f(x)
\]

Knowing symmetries of the task, for example w.r.t. a particular compact group, we can design equivariant convolutional kernels

Taco S. Cohen, Max Welling, Group Equivariant Convolutional Networks, ICML 2016
Taco S. Cohen, Max Welling, Steerable CNNs, ICLR 2017
Taco S. Cohen, Mario Geiger, Maurice Weiler, A General Theory of Equivariant CNNs on Homogeneous Spaces, NeurIPS 2019
Machine learning design for wireless communication

wireless domain knowledge → inductive bias in machine learning design

Wireless Domain Expertise

**Benefits:**
- Modeling flexibility, computational efficiency
- Out-of-domain generalization, interpretability

**Challenges:**
- Complex modeling, processing complexity
- Out-of-domain generalization, interpretability

Complementary advantages

---

Channel modeling

Positioning, SLAM

Communication design
Neural Augmentation of Kalman Filter with Hypernetwork for Channel Tracking

Kumar Pratik, Rana Ali Amjad, Arash Behboodi, Joseph Soriaga, Max Welling

Globecom 2021

Channel Tracking
A discrete time stochastic process

- The communication channel between the receiver and the transmitter keeps evolving with time.
- The underlying stochastic process is a complex function of a lot of external factors such as environmental obstructions, reflections, and the relative velocity and alignment between the transmitter and the receiver.
- At regular intervals, pilot symbols are transmitted resulting in periodic noisy observations ($o_t$) of the ground truth channel.
- The aim is to estimate and track channel ($h_t$) at all the time steps $t$.
- Analytical channel tracking models fail to capture complex dynamic scenarios accurately.
AR-2 Kalman filter based Channel Tracking

- Optimal Kalman filter parameters vary with Doppler values
- A single Kalman filter should not be used for all the Doppler values
- The aim is to track channels following multiple different dynamics
- Channel profile: CDL-B

\[
h_t = F_t^1 h_{t-1} + F_t^2 h_{t-2} + w_t \\
o_t = H_t h_t + v_t
\]

NN Baseline
LSTM based channel tracking

• What can a standalone NN achieve?
• Single layer LSTM used as RNN
• Hidden state size \( = 2 \times \text{num. taps tracked} \)
• Synthetic observations to counter sporadically available inputs (pilots)
• We use real + imaginary representation of complex numbers for PyTorch
• Loss function: \( \sum_{t=1}^{1500} \text{MSE}(h_t, \tilde{h}_t) + \text{MSE}(o_t, \tilde{o}_t) \)
Hypernetwork based Kalman filter (HKF)

LSTM as Hypernetwork, at every step the LSTM updates the KF with the optimal set of parameters

- $F_t^1, F_t^2$: Transition matrix
- $Q_t$: Process noise covariance, $w_t \sim \mathcal{N}(0, Q_t)$
- $H_t$: Observation matrix (Identity in our case)
- $R_t$: Observation noise covariance (SNR), $v_t \sim \mathcal{N}(0, R_t)$

Here, we don’t need to model observation dynamics parameters as it is same for every Doppler scenario.

- $H_t = \mathbb{I}$ (Identity for our case)
- $R_t = \sigma_{SNR}^2 \cdot \mathbb{I}$ (determined by SNR and is provided externally)

Hypermnetwork based Kalman filter

Residual in base Kalman filter parameters

**Autoregressive-2 KF (AR-2)**

\[ h_t = F^1_t h_{t-1} + F^2_t h_{t-2} + w_t \]

\[ \alpha_t = H_t h_t + v_t \]

- \( F^1_t, F^2_t \) = Transition matrix
- \( Q_t \) = Process noise covariance

- \( H_t \) = Observation matrix (Identity in our case)
- \( R_t \) = Observation noise covariance

\[ w_t \sim \mathcal{N}(0, Q_t) \]
\[ v_t \sim \mathcal{N}(0, R_t) \]

\[ \theta = \{ F^1, F^2, Q \} \]
- \( F^1 = \mathbb{I} \) (Identity)
- \( F^2 = 0 \) (Zero)
- \( Q \) = Analytic Kalman \( Q \)

\[ \Delta \theta_{t+1} = \{ \Delta F^1_t, \Delta F^2_t, \Delta Q_t \} \]

- \( F^1_t = F^1 + \Delta F^1_t \)
- \( F^2_t = F^2 + \Delta F^2_t \)
- \( Q_t = Q + \Delta Q_t \)
Hypernetwork based Kalman filter

What happens in case of missing observation?

- $F^1_t, F^2_t = \text{Transition matrix}$
- $Q_t = \text{Process noise covariance}$
- $H_t = \mathbb{I} = \text{Observation matrix (Identity in our case)}$
- $R_t = R = \sigma^2_{SNR} \cdot \mathbb{I} = \text{Observation noise covariance}$

Reparameterization trick in case of missing observations

In case we don’t have observations, we can sample its value:

- $\hat{\theta}_{t+1} \sim \mathcal{N}(\hat{h}_{t+1}, R)$  \[\text{Reparameterization trick}^*\]
- $\hat{\theta}_{t+1} = \hat{h}_{t+1} + \epsilon \odot R_{diag}^{1/2}, \epsilon \sim \mathcal{N}(0, \mathbb{I})$

Training Loss $= \sum_{m=1}^{M} \sum_{t}^{T=1500} \text{MSE}(h_t, \hat{h}_t(\psi))$

Simulation parameters

- Channel profile: CDL-B, SISO setup
- Delay spread: 100 ns
- FFT size (number of subcarriers): 4096
- Modulation: QPSK
- Subcarrier spacing: 30 kHz
- Carrier frequency: 4 GHz
- SNR = 10 dB
- Pilot ratio = 1:6
- Each channel is 1500 OFDM symbols long, i.e., 1500 timesteps in each sequence
- We are tracking 64 channel taps (time domain channel tracking) and report NMSE: $\mathbb{E} \left[ \frac{||h_t - h_t||^2_2}{||h_t||^2_2} \right]$
Experimental setup

Configuration and Dataset

- We have five separate bins with each bin having three Doppler values
- Each Doppler value has 800 training channel instances, and 200 validation/test channel instances
- Below we mention the 15 Doppler values (and their corresponding velocities) classified into respective bins

<table>
<thead>
<tr>
<th>Doppler bins</th>
<th>Doppler values in the bin (Hz)</th>
<th>corresponding velocities in km/hr</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bin 0 (0 – 70 Hz)</td>
<td>0, 30, 60</td>
<td>0, 8, 16</td>
</tr>
<tr>
<td>Bin 1 (70 - 150 Hz)</td>
<td>70, 100, 130</td>
<td>18, 27, 35</td>
</tr>
<tr>
<td>Bin 2 (150 – 300 Hz)</td>
<td>150, 210, 270</td>
<td>40.5, 56.6, 72.8</td>
</tr>
<tr>
<td>Bin 3 (300 - 500 Hz)</td>
<td>300, 400, 500</td>
<td>81, 108, 135</td>
</tr>
<tr>
<td>Bin 4 (500 – 1850)</td>
<td>800, 1300, 1850</td>
<td>215.8, 350.7, 499</td>
</tr>
</tbody>
</table>
Results

<table>
<thead>
<tr>
<th>Doppler</th>
<th>GKF</th>
<th>BKF</th>
<th>LSTM</th>
<th>HKF</th>
<th>HKF-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 Hz</td>
<td>-48.89</td>
<td>-31.78</td>
<td>-18.05</td>
<td>-29.99</td>
<td>-31.86</td>
</tr>
<tr>
<td>30 Hz</td>
<td>-32.60</td>
<td>-32.60</td>
<td>-22.16</td>
<td>-30.59</td>
<td>-30.62</td>
</tr>
<tr>
<td>60 Hz</td>
<td>-31.40</td>
<td>-28.47</td>
<td>-26.77</td>
<td>-30.76</td>
<td>-30.92</td>
</tr>
<tr>
<td>70 Hz</td>
<td>-30.64</td>
<td>-29.84</td>
<td>-26.63</td>
<td>-30.75</td>
<td><strong>-30.95</strong></td>
</tr>
<tr>
<td>100 Hz</td>
<td>-27.71</td>
<td><strong>30.06</strong></td>
<td>-29.15</td>
<td>-30.80</td>
<td><strong>-31.04</strong></td>
</tr>
<tr>
<td>130 Hz</td>
<td>-28.89</td>
<td>26.96</td>
<td>-29.23</td>
<td>-30.82</td>
<td><strong>-31.22</strong></td>
</tr>
<tr>
<td>150 Hz</td>
<td>-29.64</td>
<td>-29.91</td>
<td>-29.30</td>
<td>-30.65</td>
<td><strong>-31.04</strong></td>
</tr>
<tr>
<td>210 Hz</td>
<td>-31.76</td>
<td>-30.76</td>
<td>-29.19</td>
<td>-30.62</td>
<td>-30.80</td>
</tr>
<tr>
<td>270 Hz</td>
<td>-30.66</td>
<td>-28.61</td>
<td>-29.12</td>
<td>-30.33</td>
<td>-30.44</td>
</tr>
<tr>
<td>300 Hz</td>
<td>-29.68</td>
<td><strong>30.18</strong></td>
<td>-29.27</td>
<td>-30.20</td>
<td><strong>-30.22</strong></td>
</tr>
<tr>
<td>400 Hz</td>
<td>-30.24</td>
<td>-29.98</td>
<td>-28.15</td>
<td>-29.48</td>
<td>-29.38</td>
</tr>
<tr>
<td>800 Hz</td>
<td>-26.70</td>
<td><strong>-18.73</strong></td>
<td>-25.59</td>
<td>-26.47</td>
<td>-26.55</td>
</tr>
<tr>
<td>1300 Hz</td>
<td>-21.65</td>
<td><strong>-17.53</strong></td>
<td>-22.01</td>
<td>-22.85</td>
<td><strong>-23.24</strong></td>
</tr>
<tr>
<td>1850 Hz</td>
<td>-16.86</td>
<td><strong>-15.25</strong></td>
<td>-18.29</td>
<td>-19.18</td>
<td><strong>-19.67</strong></td>
</tr>
</tbody>
</table>

- **Global**: One model trained over entire range of Doppler values
- **Binned KF**: KF params computed over Doppler values in the bin
- **Genie KF**: Analytically computed Kalman parameters per Doppler
- **LSTM**: Vanilla LSTM baseline trained over entire range of Dopplers
- **HKF**: Hypernetwork Kalman filter trained over entire range of Dopplers

- **Bin 0 (0-70 Hz)**: 0, 30, 60 Hz
- **Bin 1 (70-150 Hz)**: 70, 100, 130 Hz
- **Bin 2 (150-300 Hz)**: 150, 210, 270 Hz
- **Bin 3 (300-500 Hz)**: 300, 400, 500 Hz
- **Bin 4 (500-1850 Hz)**: 800, 1300, 1850 Hz
### Results

**Evaluation on untrained Pilot ratios (Pilot ratio: 1:3, Pilot ratio: 1:6)**

Test SNR: 10 dB

<table>
<thead>
<tr>
<th>NMSE (in dB)</th>
<th>Seen Doppler values</th>
<th>Unseen Doppler values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Doppler</strong></td>
<td>BKF</td>
<td>LSTM</td>
</tr>
<tr>
<td>0 Hz</td>
<td>-35.33</td>
<td>-18.63</td>
</tr>
<tr>
<td>30 Hz</td>
<td>-34.80</td>
<td>-20.64</td>
</tr>
<tr>
<td>60 Hz</td>
<td>-31.33</td>
<td>-20.21</td>
</tr>
<tr>
<td>70 Hz</td>
<td>-31.96</td>
<td>-20.16</td>
</tr>
<tr>
<td>100 Hz</td>
<td>-33.04</td>
<td>-18.67</td>
</tr>
<tr>
<td>130 Hz</td>
<td>-30.48</td>
<td>-17.27</td>
</tr>
<tr>
<td>150 Hz</td>
<td>-33.09</td>
<td>-16.16</td>
</tr>
<tr>
<td>210 Hz</td>
<td>-33.40</td>
<td>-14.50</td>
</tr>
<tr>
<td>270 Hz</td>
<td>-32.19</td>
<td>-13.18</td>
</tr>
<tr>
<td>300 Hz</td>
<td>-33.19</td>
<td>-12.65</td>
</tr>
<tr>
<td>400 Hz</td>
<td>-32.90</td>
<td>-11.33</td>
</tr>
<tr>
<td>500 Hz</td>
<td>-32.26</td>
<td>-10.57</td>
</tr>
<tr>
<td>800 Hz</td>
<td>-28.06</td>
<td>-10.52</td>
</tr>
<tr>
<td>1850 Hz</td>
<td>-25.30</td>
<td>-7.60</td>
</tr>
</tbody>
</table>

**TABLE IV** Evaluation on untrained pilot ratio of 1 : 3

<table>
<thead>
<tr>
<th>NMSE (in dB)</th>
<th>Seen Doppler values</th>
<th>Unseen Doppler values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Doppler</strong></td>
<td>BKF</td>
<td>LSTM</td>
</tr>
<tr>
<td>0 Hz</td>
<td>-28.56</td>
<td>-4.55</td>
</tr>
<tr>
<td>30 Hz</td>
<td>-31.70</td>
<td>-5.55</td>
</tr>
<tr>
<td>60 Hz</td>
<td>-25.80</td>
<td>-4.85</td>
</tr>
<tr>
<td>70 Hz</td>
<td>-26.16</td>
<td>-4.31</td>
</tr>
<tr>
<td>100 Hz</td>
<td>-27.59</td>
<td>-3.29</td>
</tr>
<tr>
<td>130 Hz</td>
<td>-23.34</td>
<td>-2.11</td>
</tr>
<tr>
<td>150 Hz</td>
<td>-26.00</td>
<td>-1.34</td>
</tr>
<tr>
<td>210 Hz</td>
<td>-27.90</td>
<td>-0.32</td>
</tr>
<tr>
<td>270 Hz</td>
<td>-24.41</td>
<td>0.34</td>
</tr>
<tr>
<td>300 Hz</td>
<td>-26.77</td>
<td>0.54</td>
</tr>
<tr>
<td>400 Hz</td>
<td>-26.86</td>
<td>1.07</td>
</tr>
<tr>
<td>500 Hz</td>
<td>-24.77</td>
<td>1.34</td>
</tr>
<tr>
<td>800 Hz</td>
<td>-10.36</td>
<td>1.95</td>
</tr>
<tr>
<td>1300 Hz</td>
<td>-9.50</td>
<td>2.19</td>
</tr>
<tr>
<td>1850 Hz</td>
<td>-7.25</td>
<td>2.23</td>
</tr>
</tbody>
</table>

**TABLE V** Evaluation on untrained pilot ratio of 1 : 10

- **Bin 0 (0-70 Hz): 0, 30, 60 Hz**
- **Bin 1 (70-150 Hz): 70, 100, 130 Hz**
- **Bin 2 (150-300 Hz): 150, 210, 270 Hz**
- **Bin 3 (300-500 Hz): 300, 400, 500 Hz**
- **Bin 4 (500-1850 Hz): 800, 1300, 1850 Hz**

- **Global**: One model trained over entire range of Doppler values
- **Binned KF**: KF params computed over Doppler values in the bin
- **Genie KF**: Analytically computed Kalman parameters per Doppler
- **LSTM**: Vanilla LSTM baseline trained over entire range of Dopplers
- **HKF**: Hypernetwork Kalman filter
Conclusion

• The proposed Hypernetwork-KF (HKF) combines the robustness of Kalman filter (KF) with the expressive power of neural networks (NN)

• A single Hypernetwork-KF (HKF) can track channels following multiple different dynamics

• Solely LSTM based baseline shows poor generalization when tested on settings different from the training data

• Through extensive experiments, we showed that our hybrid HKF outperforms the standalone KF or NN based methods

• Interpretability and out-of-domain generalization with Neural augmentation
MIMO-GAN: Generative MIMO Channel Modeling

Tribhuvanesh Orekondy, Arash Behboodi, Joseph Soriaga
ICC 2022
https://arxiv.org/abs/2203.08588
**Neural Channel Modelling**

- **Channel Modelling**
  - Model physical propagation effects on wireless signals

- **Existing Channel Models**
  - Standard Channel Models: 3GPP TDL/CDL, WINNER, ...
  - Ray Tracing

- **Building a (classical) channel model is challenging**
  - Domain expertise
  - Cumbersome field measurements
  - Hard-coded assumptions
  - Limited scenarios, Slow to prototype

- **Our goal**: Data-driven neural channel models
  + Accurately match field data distribution
Problem Statement

Given I/O measurements:
\[ \{(x_i, y_i)\}_{i=1}^{N} \]

Learn parameters of channel model:
\[ y = h_\theta(x) \]
Related Work

Input/Output symbols

Simple Channels (e.g., AWGN)

Explicitly models $p(y|x)$

SISO


MIMO-GAN: Approach

\[ z \sim \mathcal{N}(0, I) \]

\[ (i, j) \rightarrow \text{embed} \rightarrow G \rightarrow h_{i,j}(\tau) \in \mathbb{C}^L \]

\[ \mathbf{x} \rightarrow \text{Channel} \rightarrow \mathbf{x} \star \mathbf{H}_{gt} \]

\[ \mathbf{y} \rightarrow \hat{\mathbf{y}} \rightarrow \hat{\mathbf{y}}^H \hat{\mathbf{y}} \rightarrow \mathbf{D} \rightarrow \text{real/fake} \]

Training objective: WGAN-GP

\[
\max_G \min_D \mathbb{E}_{\hat{\mathbf{y}} \sim p_G} [D(\hat{\mathbf{y}}, \hat{\mathbf{y}}^H \hat{\mathbf{y}})] - \mathbb{E}_{\mathbf{y} \sim p_{data}} [D(\mathbf{y}, \mathbf{y}^H \mathbf{y})] + \lambda \mathbb{E}_{\hat{\mathbf{y}} \sim p_\mathbf{y}} \left( \left\| \nabla_{\hat{\mathbf{y}}} D(\hat{\mathbf{y}}, \hat{\mathbf{y}}^H \hat{\mathbf{y}}) \right\|^2 \right) \\
\hat{\mathbf{y}} = \mathbf{x} \star G(z) \]
Evaluation: Setting

- Channel
  - TDL-A and TDL-B
  - 4×4 channels
  - Delay spread: 300 ns

- Dataset
  - Transmit signals $x = $ Digital impulse
  - 60k input-output measurements
Evaluation: Power and Delay Profile

![Graphs showing power and delay profile for TDL-A with Rx = 4](image)

### TABLE I: Power and delay statistics of MIMO-GAN and ground-truth (GT) channels.

<table>
<thead>
<tr>
<th></th>
<th>Total Power (dB)</th>
<th>Average Delay (µs)</th>
<th>RMS Delay Spread (µs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TDL-A</td>
<td>MIMO-GAN</td>
<td>4.648</td>
<td>0.2643</td>
</tr>
<tr>
<td></td>
<td>GT</td>
<td>4.628</td>
<td>0.2641</td>
</tr>
<tr>
<td></td>
<td>MAE</td>
<td>-18.69</td>
<td>3.57×10^{-3}</td>
</tr>
<tr>
<td>TDL-B</td>
<td>MIMO-GAN</td>
<td>4.735</td>
<td>0.2276</td>
</tr>
<tr>
<td></td>
<td>GT</td>
<td>4.688</td>
<td>0.2285</td>
</tr>
<tr>
<td></td>
<td>MAE</td>
<td>-14.99</td>
<td>3.37×10^{-3}</td>
</tr>
</tbody>
</table>

-18 dB MAE

3.57 ns MAE
Evaluation: Condition vs. Unconditional Generation

- Q: Do we need to condition networks on spatial co-ordinates?

```
\begin{align*}
\mathbf{z} & \sim \mathcal{N}(0, \mathbf{I}) \\
(i, j) & \xrightarrow{\text{embed}} \mathbf{G} \\
& \quad \mathbf{h}_{i,j}(\mathcal{T}) \in \mathbb{C}^L \\
& \quad \left( \begin{array}{c} \hat{\mathbf{y}}_i \\ \hat{\mathbf{y}}_i^H \end{array} \right) \\
& \quad \left( \begin{array}{c} \mathbf{y}_i \\ \mathbf{y}_i^H \end{array} \right)
\end{align*}
```

\[
i = 1, \ldots, N_R \\
j = 1, \ldots, N_T
\]

**TABLE II:** MAEs of Power and Delay statistics comparing *unconditioned* (`X`) and *conditioned* (`✓`) generation/discrimination of channels. Last row corresponds to MIMO-GAN. In each column, we represent the best performance in **bold**.

<table>
<thead>
<tr>
<th>Cond. G</th>
<th>Cond. D</th>
<th>Total power (dB)</th>
<th>Avg. Delay (ns)</th>
<th>RMS Spread (ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>X</td>
<td>-11.47</td>
<td>49.00</td>
<td>82.00</td>
</tr>
<tr>
<td>X</td>
<td>✓</td>
<td>-11.67</td>
<td>35.10</td>
<td>68.00</td>
</tr>
<tr>
<td>✓</td>
<td>X</td>
<td>-12.25</td>
<td>3.76</td>
<td>7.93</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td><strong>-18.73</strong></td>
<td><strong>0.24</strong></td>
<td><strong>3.57</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Total power (dB)</th>
<th>Avg. Delay (ns)</th>
<th>RMS Spread (ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.13</td>
<td>105.20</td>
<td>152.8</td>
</tr>
<tr>
<td>-6.03</td>
<td>82.94</td>
<td>133.4</td>
</tr>
<tr>
<td>-11.74</td>
<td>18.72</td>
<td>23.01</td>
</tr>
<tr>
<td><strong>-14.95</strong></td>
<td><strong>0.90</strong></td>
<td><strong>3.37</strong></td>
</tr>
</tbody>
</table>
Evaluation: Spatial Correlation

\[ MAE(R_{\text{GT}}, R_{\text{MIMO-GAN}}) \]

<table>
<thead>
<tr>
<th>GM</th>
<th>SQ</th>
<th>TDL-A</th>
<th>TDL-B</th>
</tr>
</thead>
<tbody>
<tr>
<td>✗</td>
<td>✗</td>
<td>0.144</td>
<td>0.171</td>
</tr>
<tr>
<td>✓</td>
<td>✗</td>
<td>0.204</td>
<td>0.061</td>
</tr>
<tr>
<td>✗</td>
<td>✓</td>
<td>0.071</td>
<td>0.177</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>0.059</td>
<td>0.057</td>
</tr>
</tbody>
</table>

- “GM”
  - Condition discriminator additionally on receive gram matrices: \( D(y, y^H y) \)
- “SQ”
  - Sequential impulses across transmit antennas

\[ R_{TX} = \mathbb{E}[H^H H] \]

(TDL-A) MAE: 0.059

\[ R_{RX} = \mathbb{E}[HH^H] \]

(TDL-A) MAE: 0.057
Conclusion: MIMO-GAN

• Learning complex distributions that are easy to sample from

• Model adaptation: it can be adapted to variable number of antenna

• Learning distribution: domain specific crafted features for discriminator networks
  - Challenge: evaluating generative models
Neural RF SLAM
Indoor unsupervised positioning and mapping of CSI
Shreya Kadambi, Arash Behboodi, Joseph Soriaga, Max Welling, Roohollah Amiri, Srinivas Yerramalli, Taesang Yoo
ICC 2022
https://arxiv.org/abs/2203.08264
Positioning and Mapping problem

We have one (or multiple) anchors (APs, gNBs)

- **Goal 1**: find the location of user
- **Goal 2**: find the map of environment (reflectors, etc.)

**Classical solutions:**

- **Triangulation/trilateration:**
  - knowing anchor locations, find UE location based on mutual distance/angle measurement
  - Common features: angle of arrival (AoA), time of flight (ToF), time difference of arrival (TDoA)

- **Fingerprinting:**
  - Data driven solution: field data of (feature, location) + train an ML algorithm (kNN, neural networks, etc.)
Unsupervised approach

• **Idea 1:** if you have enough multi-path components (reflection, scatterers), we can localize even with a single anchor

• **Idea 2:** with enough unlabeled CSI samples, we can learn the geometry of the environment without labels (location information)

• **Assumption:** We have access to many unlabeled user traces (CSI $H_{u,k}$, ToF $\{\tau_k\}_u$, TDOA $\{\Delta\tau_k\}_u$, AoA $\{\phi_i, \theta_i\}_u$)
Unsupervised approach

- Reflectors can be modelled using a single **virtual anchor** obtained by reflection of the main anchor*
- ToF/AoA can be obtained as
  \[
  \text{TOF: } \tau_1 = \frac{||p-p_1||}{c}, \text{ AoA: } \theta_1 = \arctan \frac{x-x_1}{y-y_1}
  \]

Notation:
- \(p_0\): anchor location
- \(p_i\): virtual anchor location
- \(p_u\): location of the user \(u\) for \(u \in \mathcal{U}\)
- \(\tau_{i,u}\): time of flight for path \(i\) of user \(u\)

Problem: find \(p_u\)’s and \(p_i\)’s from \(\{\tau_{i,u}, i \in I\}\)

*similar VA based modeling can be done for other effects
Neural RF SLAM architecture

• For a proper choice of loss function \( \ell(.) \), solve:
\[
\min \sum_{u \in U} \ell \left( \{ \tau_{i,u} \mid i \in I_u \}, \left\{ \frac{\| p_u - p_i \|}{c}, i \in \{0, \ldots, N_{VA}\} \right\} \right)
\]

• Parameterize the environment using \( p_i \) with \( p_0 \) being the anchor and \( p_i \)'s being virtual anchors \( (i \neq 0) \)

• Replace \( p_u \) by a neural network \( g_W(H_u) \):
\[
\text{arg} \min_{W, p_1, \ldots, p_{N_{VA}}} \sum_{u \in U} \ell \left( \{ \tau_{i,u} \mid i \in I_u \}, \left\{ \frac{\| g_W(H_u) - p_i \|}{c}, i \in \{0, \ldots, N_{VA}\} \right\} \right)
\]

\( \ell(.) \): should be a set prediction loss (Chamfer, Hungarian, etc.)
Association problem

• With only ToFs, we have to first associate each delay $\tau_i$ with a virtual anchor $VA_k$
• We treat it as a set prediction problem
• $g_W(\cdot)$ should act on the set of ToFs (permutation invariant)
• The loss function $\ell(\cdot)$ should act on two sets
  • Explored Chamfer, Hausdorff and Hungarian set loss
• We use Hungarian algorithm to match two sets first

Thought experiment let's assume we know virtual anchor locations
A user with unknown location receives the signal with the delays $\tau_0 \leq \tau_1 \leq \tau_2$

$\tau_0$: the smallest ToF corresponds to the main anchor
How do we associate $\tau_1$ and $\tau_2$ to $VA_1$ and $VA_2$
Isometric ambiguities

- ToF profiles and CSI are invariant to isometric transformation (rotation, translation and reflection)

**Post-training correction:**
- Detach and fix the localization network
- Remove the ambiguity with few reference points
- Linear map (2 × 2 or 3 × 3) can be used to correct the mapping part

NOTE: Same correction is applied to ToF/TDoA SLAM as well
2D Dataset

Propagation effect:
- Single bounce reflection, internally implemented ray-tracer
- Path i: \((a_i, \tau_i, \theta_i)\) - the path is determined from the propagation environment

Models:
Supervised Localization, Supervised Mapping, Neural SLAM (MLP, DeepSet, ConvNet)

Modalities:
Time Difference of Arrival - TDoA - \(\Delta \tau_{u,i}\), Channel State Information - \(H_{u,k}\)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>2D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Carrier frequency</td>
<td>2GHz</td>
</tr>
<tr>
<td>Bandwidth</td>
<td>400Mhz</td>
</tr>
<tr>
<td>Test area</td>
<td>5mx5m</td>
</tr>
<tr>
<td>Num of subcarriers</td>
<td>128</td>
</tr>
<tr>
<td>Number of walls</td>
<td>4</td>
</tr>
</tbody>
</table>

Source ample text
Experiments for 2D SLAM

2D Dataset for bandwidth of 400Mhz

- Features extracted using MUSIC algorithm and overparametrized VA numbers
- End-to-end fully unsupervised SLAM

<table>
<thead>
<tr>
<th>Localization error</th>
<th>Genie TDoAs (m)</th>
<th>MUSIC TDoAs (m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average</td>
<td>0.01</td>
<td>0.154</td>
</tr>
<tr>
<td>Median</td>
<td>0.01</td>
<td>0.133</td>
</tr>
<tr>
<td>90 quantile</td>
<td>0.023</td>
<td>0.26</td>
</tr>
</tbody>
</table>

Room layout and placement of VA:
- Predicted virtual anchors in blue,
- True virtual anchors in green,
- Training area marked in red box,
- Room dimensions marked as blue box,
- TX/Anchor location in red

log norm(CSI) point cloud per tone
left: GT  right: converged model
Experiments on 3D data

Propagation effect:
- Double bounce reflection

RemCom dataset:
- 3D raytracing simulator
- Supports non smooth walls and diffraction and diffused scattering models
- Multiple reflection bounces up to orders of 6, multiple rooms and floors

Modalities:
- Time of Flight - ToF = $\tau_{u,i}$, TDoA - $\Delta\tau_{u,i}$, Channel State Information - $H_{u,k}$

<table>
<thead>
<tr>
<th>Parameter</th>
<th>3D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Carrier frequency</td>
<td>3.5Ghz</td>
</tr>
<tr>
<td>Bandwidth</td>
<td>100Mhz</td>
</tr>
<tr>
<td>Test area</td>
<td>30mx20mx4m</td>
</tr>
<tr>
<td>Num of subcarriers</td>
<td>128</td>
</tr>
<tr>
<td>Number of walls</td>
<td>6 - 7</td>
</tr>
</tbody>
</table>
Experiments on 3D SLAM: Neural SLAM on single bounce

- With 100 MHz BW, not all ToFs can be recovered from CSI values, which causes performance degradation in Neural SLAM.
- With larger environments, it is more challenging to train a model for the whole room → it is easier to focus on smaller area for neural SLAM

<table>
<thead>
<tr>
<th>Localization error</th>
<th>Median</th>
<th>90% quantile</th>
</tr>
</thead>
<tbody>
<tr>
<td>3D Neural genie ToF SLAM</td>
<td>3.4 cm</td>
<td>7 cm</td>
</tr>
<tr>
<td>3D Neural MUSIC ToF SLAM</td>
<td>43.4 cm</td>
<td>1.2m</td>
</tr>
</tbody>
</table>
Conclusion: Neural RF SLAM

- Incorporate physics of propagation in the model (decoder part)
- Unsupervised learning: unlabeled CSI samples
- Model adaptation: variable size inputs with permutation invariant functions using DeepSets
WiCluster: Passive Indoor WiFi Positioning without Precise Labels


“Modality-Agnostic Topology Aware Localization”, Farhad Ghazvinian Zanjani · Ilia Karmanov · Hanno Ackermann · Daniel Dijkman · Simone Merlin · Max Welling · Fatih Porikli, NeurlPS 2021 https://openreview.net/forum?id=3v6n7458GAg
Passive positioning

Environment No. 1 - 2D Office, 14m x 20m

Environment No. 2 - 2D Office, 15m x 21m

Environment No. 3 - 3D Home
Motion detected:
Lab

Positioning Accuracy
Cumulative Distribution Function

Position Prediction Accuracy
Absolute Error (cm)
Final points
Machine learning design for wireless communication

• Unsupervised learning:
  o Learning distributions and manifolds is an approach to obtain features in an unsupervised way
  o Examples: WiClustering, Neural RF SLAM
  o Other perspectives: self-supervised learning, transfer learning

• Adaptive models:
  o Models should be able to adapt to different channel conditions and setups
  o Examples: Hypernetwork Kalman, MIMO GAN

• Generalization
  o Designing ML models based on inductive bias, gained from domain knowledge, or neural augmentation can help generalization
  o Example: Hypernetwork Kalman, MIMO-GAN

• Interpretability
  o Neural augmentation helps interpretability of modules in an ML model
  o Examples: Hypernetwork Kalman, MIMO-GAN
Nothing in these materials is an offer to sell any of the components or devices referenced herein.

©2018-2022 Qualcomm Technologies, Inc. and/or its affiliated companies. All Rights Reserved.

Qualcomm is a trademark or registered trademark of Qualcomm Incorporated. Other products and brand names may be trademarks or registered trademarks of their respective owners.

References in this presentation to “Qualcomm” may mean Qualcomm Incorporated, Qualcomm Technologies, Inc., and/or other subsidiaries or business units within the Qualcomm corporate structure, as applicable. Qualcomm Incorporated includes our licensing business, QTL, and the vast majority of our patent portfolio. Qualcomm Technologies, Inc., a subsidiary of Qualcomm Incorporated, operates, along with its subsidiaries, substantially all of our engineering, research and development functions, and substantially all of our products and services businesses, including our QCT semiconductor business.

Follow us on:  

For more information, visit us at: 
qualcomm.com & qualcomm.com/blog
Variable input dimension using Deep Sets

- ToF, TDoA and AoA are some of the modalities contained in CSI that are correlated to the UE position.
- Problem: variable sized $\{\tau_i\}_u, \{\Delta\tau_i\}_u, \{\phi_i, \theta_i\}_u$ (limited resolution and unresolvable paths)

Functions on sets has a representation of the form $\hat{f}(\tau) = \rho(\Sigma_{\tau} \phi(\tau))$

Deep Sets

**Theorem [1]:** Assuming countable set $\mathcal{X}$, a function $f: 2^\mathcal{X} \to \mathbb{R}$ is a valid set function, i.e., invariant to the permutation of elements in $\mathcal{X}$, if and only if it can be decomposed in the form $\rho(\sum_{x \in \mathcal{X}} \varphi(x))$, for suitable transformations $\rho$ and $\varphi$.

- It is also possible to build permutation equivariant model
- The proof is based on defining a bijection from $2^\mathcal{X}$ to $\mathbb{R}$ using $\sum_{x \in \mathcal{X}} 4^{-c(x)}$ with $c(x)$ is an enumeration of elements of $\mathcal{X}$
- A similar result is presented for fixed size subsets of an uncountable set $\mathcal{X}$
- Example of embedding: $\varphi(\tau_i) = e^{-j2\pi f_c \tau_i} e^{-j\frac{2\pi k}{NT_s} \tau_i} \rightarrow H_{k,m} = \sum_{i=1}^L e^{-j2\pi f_c \tau_i} e^{-j\frac{2\pi k}{NT_s} \tau_i}$ - CSI embedding as $\varphi$ and as a CNN as $\rho$