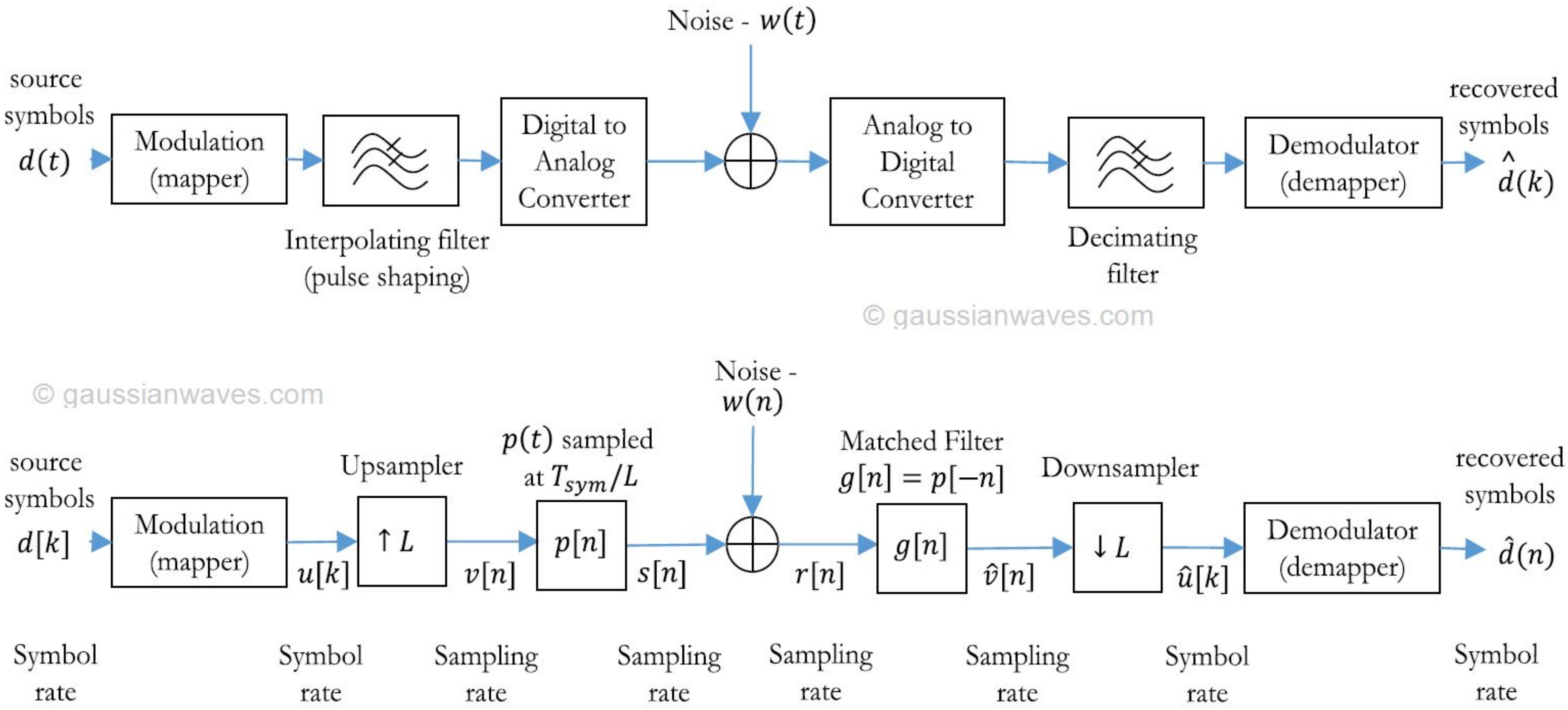
PulseMatch

Anthony Yalong

The Problem

What is the context?



The Problem

Why does this matter?

In modern communication systems, ensuring reliable signal detection and high-quality data transmission is critical, especially with the massive volume of data transmitted across various networks every day. So, with billions of devices generating continuous data, there is always the need for more reliable communication. However, when the pulse shaping function is unknown by the receiver, it becomes difficult to maximize the signal-to-noise ratio (SNR), leading to increased bit error rates and unreliable communication. Thus, there is a growing need for a solution that allows receivers to infer the optimal matched filter, improving SNR and ensuring reliable data transmission especially in these unknown scenarios. So, this project is an attempt to address this need by using deep learning techniques to train a model that can automatically learn the best pulse shaping filter for any given signal.

The Solution

What is the goal?

Pulse Shaping Inference: Be able to infer pulse shaping functions directly from received signals, without prior knowledge of the pulse shape.

Optimizing SNR: Be able to predict FIR filter taps that could maximize SNR, improving signal detection and reducing errors.

Generalizability: Be able to adapt to dynamic environments, like different modulation schemes, noise conditions, and sample per symbol rates.

The Functionality

How does it work?

Input:

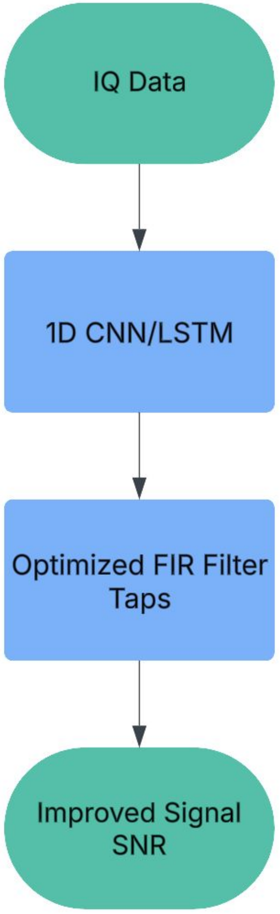
- Noisy signal in IQ format
- “Unknown” pulse shaping function applied the transmitted signal

Model:

- 1D CNN: short term dependencies
- LSTM: long & short term dependencies

Evaluation:

- Maximize SNR
    - Apply predicted FIR filter taps and determine SNR
    - SNR = Power Signal / Power Noise
- Minimize MSE
    - Predicted vs. Actual FIR filter taps



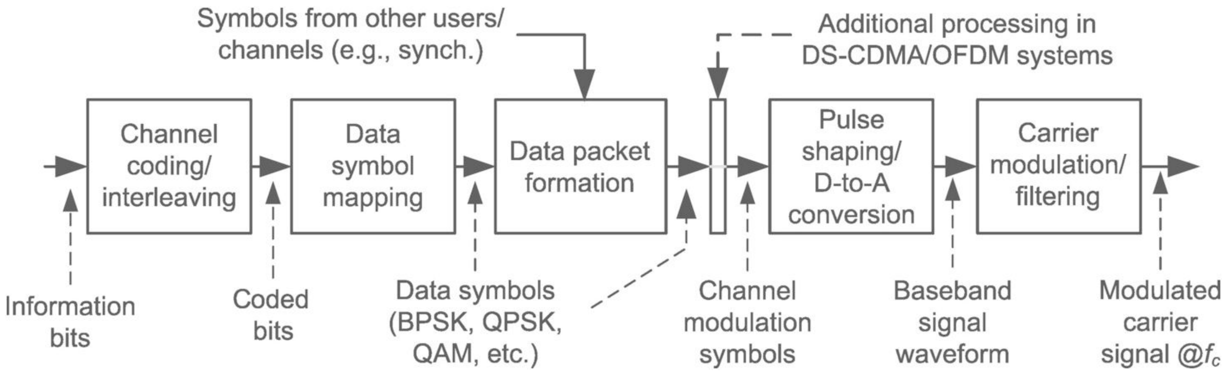
The Data

What's the input?

- Fully synthesized dataset

Data generation process:

1. Generate a random bitstream.
2. Apply a random modulation scheme (e.g., BPSK, QPSK).
3. Convolve with a random FIR filter to apply pulse shaping.
4. Set a random Samples-Per-Symbol rate.
5. Add random noise (e.g. AWGN, fading).



The Challenges/Worries

What are the expected difficulties?

Model:

- Can the model learn the FIR filter taps across different bit streams, modulation schemes, sps rates, and noise?
- Is it more useful to test the model's generalizability with just one random variable alongside the FIR filter taps?

Dataset:

- Will the dataset represent real-world data well?
- What size dataset should I use to train the model effectively?

Training:

- How can I prevent the model from overfitting to the training bit streams, modulation schemes, sps rates, and noise?

THANK YOU