### Motivation
- Entire fields hinge upon the Fourier Transform and its efficient computation
- Faster implementations of the Discrete Fourier Transform (DFT) allow for more efficient computation in a wide variety of systems, such as medical imaging, optics, and radar systems.
- Neural network architectures may be the solution to faster DFT computation times.

### Approach
- Three fully connected layers, linear activation functions
  - Training/Test Data
  - 30,000 random signals, bandlimited to 10 Hz (to avoid aliasing)
  - With/without noise
  - 90/10 training/test split

\[
\mathcal{J} = \frac{1}{m} \sum_{i=1}^{m} \| F\{x_i\} - \hat{F}\{x_i\} \|_2^2
\]

### Hyperparameter Selection
- 17 nodes per (hidden) layer
- Training epochs = 20,000
- Learning rate = 0.001
- Minibatch size = 250
- Drop-out probability = 0.9
- Other regularization was found to not improve performance

### Future Work
- Exploiting structure in signals
  - Sparsity (compressed sensing)
- Other transforms
  - Discrete Cosine transform
  - Radon transform
  - Continuous Wavelet transform

### Experimental Results
- Training Error = $8.1 \times 10^{-4}$, Test Error = $2.1 \times 10^{-2}$
- Naive DFT computation time = 4.1 $\mu$s, FFT computation time = 3.5 $\mu$s, neural network DFT computation time = 1.9 $\mu$s
- Neural network successfully estimates DFT well (see below for example)
- Empirically, architecture is 2x faster than naive computation and 1.8x faster than FFT for $N = 100$.

![Graph showing error comparison between Naive DFT and network](image)

### References
- [Link to references]
- [Link to additional resources]