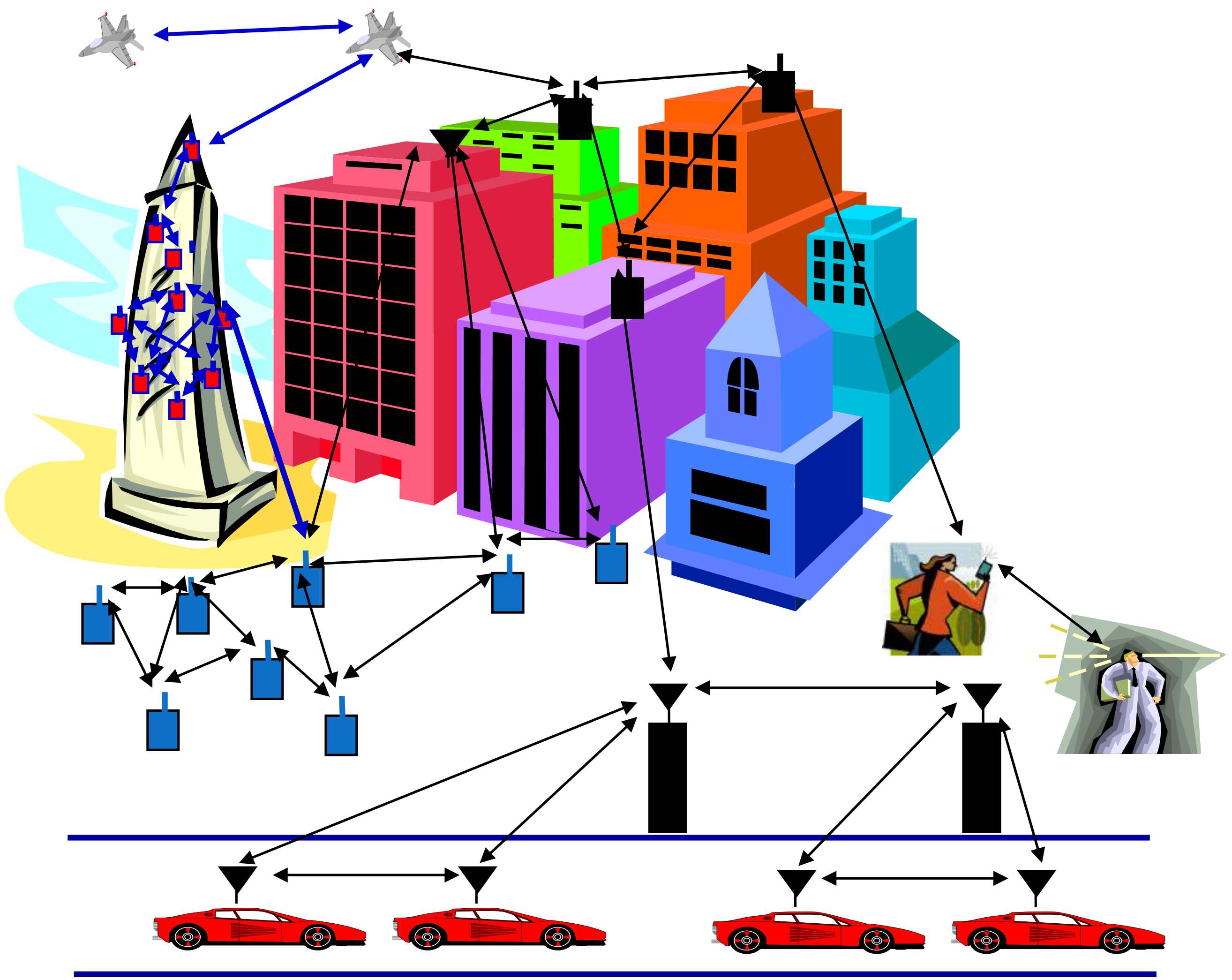
Wireless System Design using Optimization and Machine Learning

Andrea Goldsmith

ARFL/AFOSR Center of Excellence Seminar
March 8, 2021

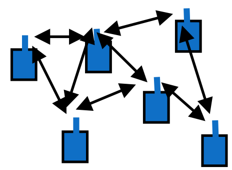
Future Wireless Networks

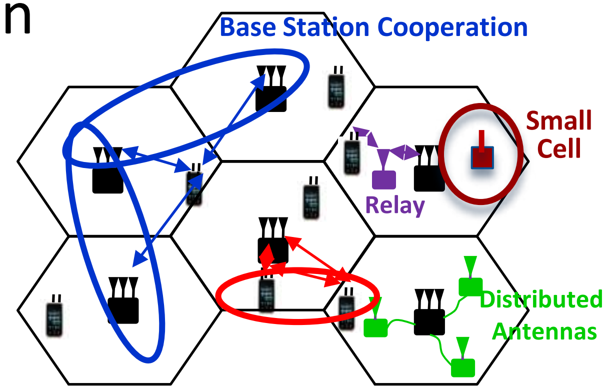
Require agility and robustness

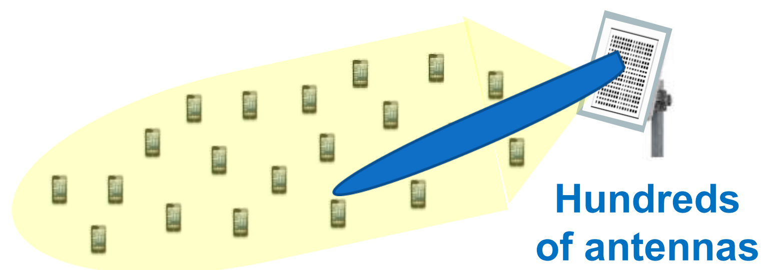


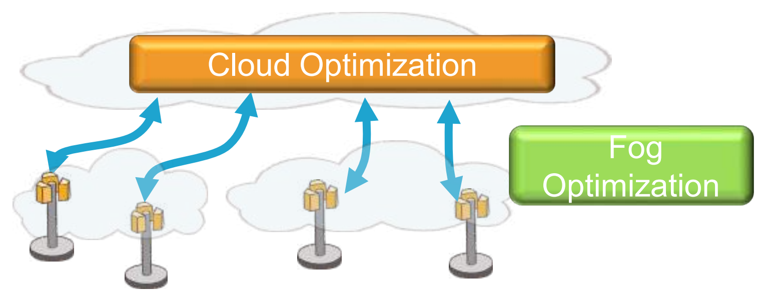
Enabling Technologies for Future Wireless Networks

- Rethinking cellular and ad-hoc network design
- Utilizing more spectrum (mmw)
- Very low power radios
- Massive MIMO (multiple antennas)
- New PHY and MAC techniques
- Multihop routing
- Edge computing and caching
- Fog optimization
- Machine learning

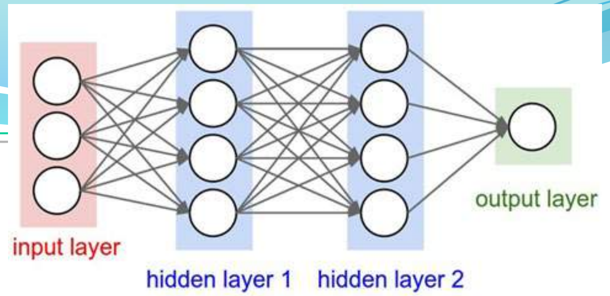






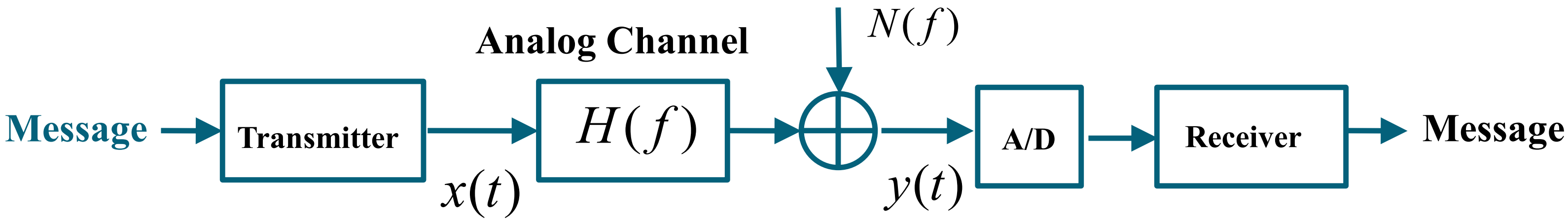


ML in Wireless Systems



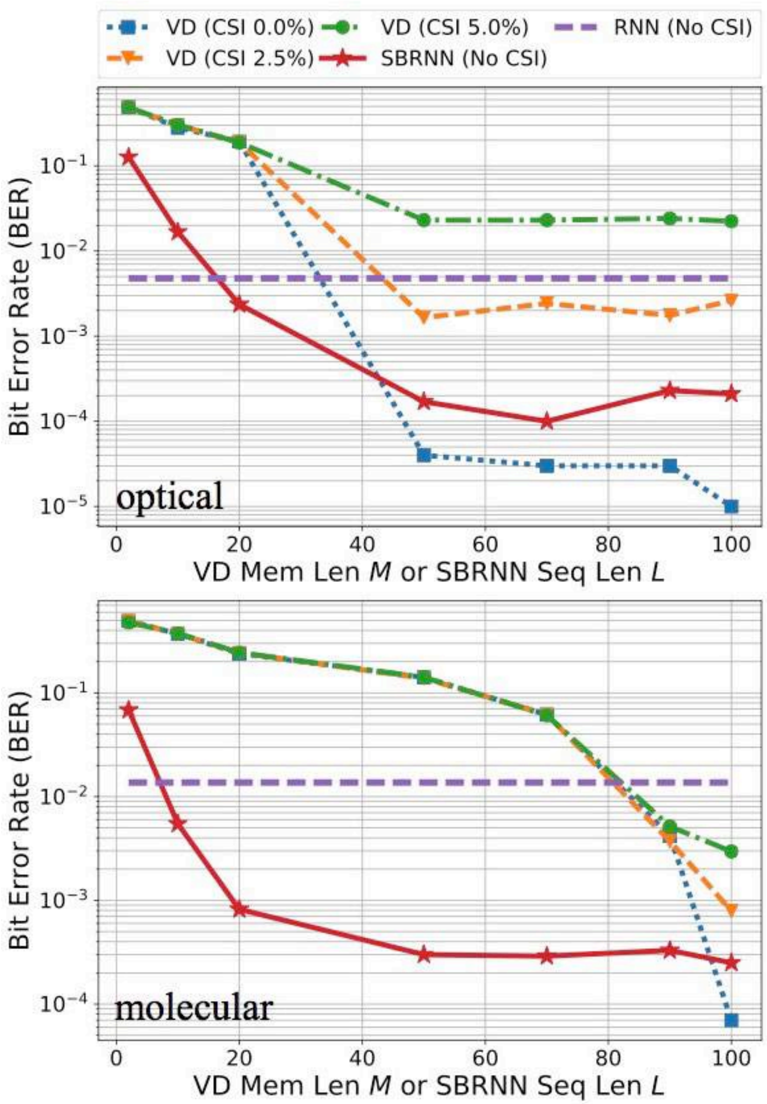
- We have shown that ML “trumps theory”:
  - In equalization of unknown/complex channels
  - In joint source and channel coding of text
- Application of ML to wireless system design
  - Detection in unknown channels (molecular, mmW, nonlinear)
  - Modulation and detection
  - Encoding and decoding
  - MIMO transmission and reception
  - Joint source and channel encoding/decoding
  - Network resource allocation
- ML algorithm and training optimization needed
  - That is where comm/network theory come in

ML in PHY layer design



- PHY transmitter and receiver design typically based on a mathematical channel mode
  - Accurate channel models may not be known
  - Models may not enable computationally efficient PHY algorithms (decoding, detection, message recovery)
- How does an ML-based approach solve this?
  - No need for an underlying model or all its parameters
  - Learn the design directly from data

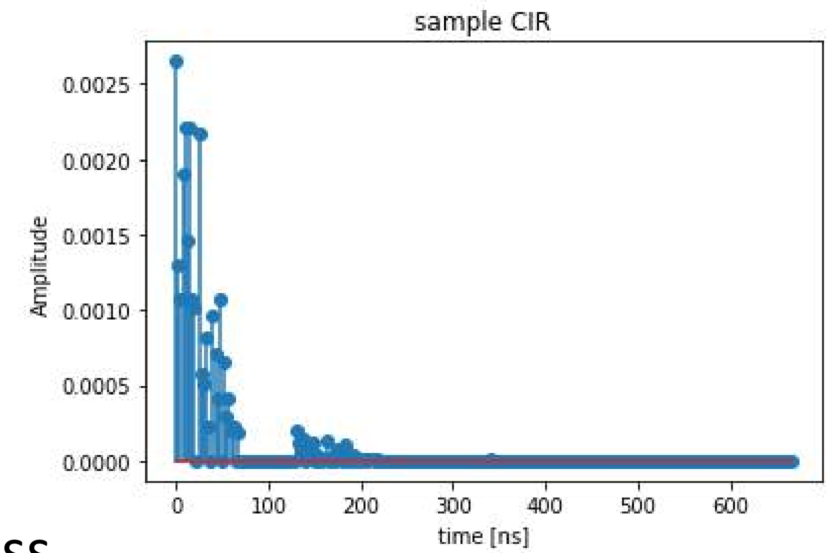
BER for Poisson/Molecular



- SBRNN outperforms VD with perfect CSI if M is less than the memory order
- SBRNN outperforms VD with CSI estimation error
- Lower M and L correspond to lower complexity in the detection algorit

mmWave Channel Model

- Channel Model*
  - Sparse time clustering
  - Long memory



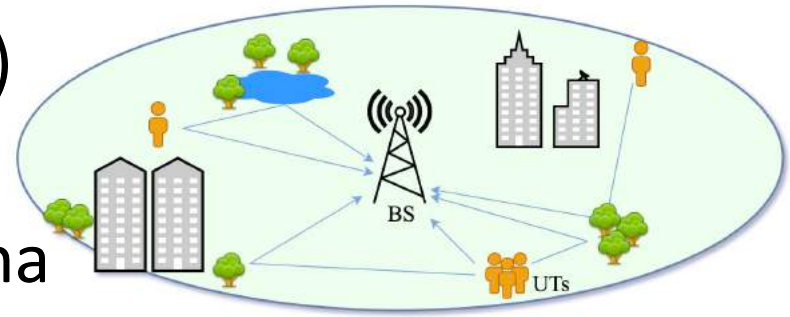
- Massive MIMO
  - Compensates for severe pathloss
  - Enables aggressive spatial reuse
  - Needs to handle highly directional links

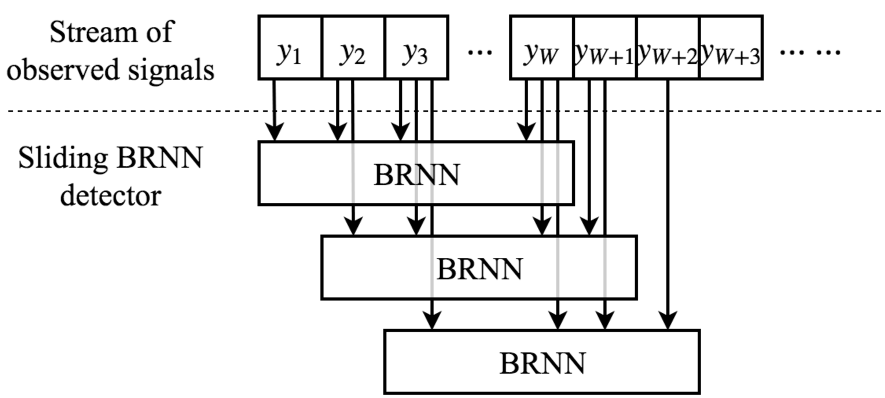
- Hard to estimate accurate CSI
- Prohibitive complexity of ML detection

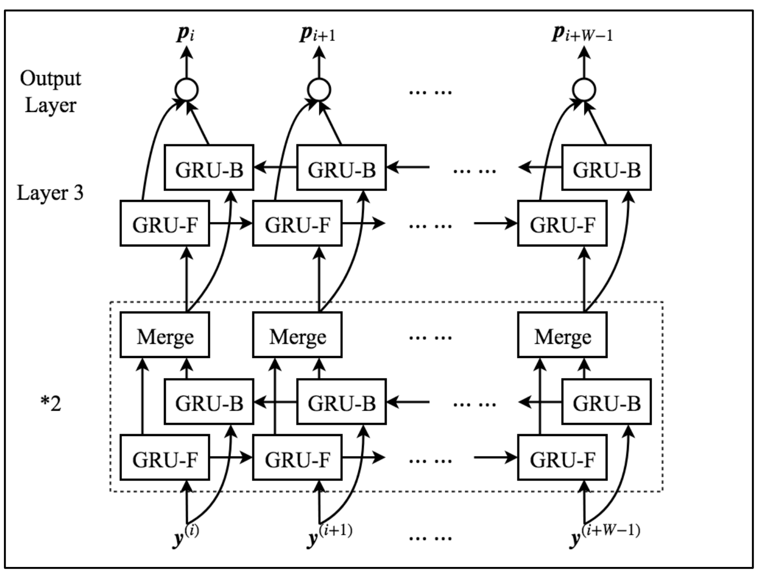
* Samimi et al. 3-D millimeter-wave statistical channel model for 5G wireless system design.

Uplink Detection with SBRNN

- Single cell mmw uplink (sparse)
- The BS has multiple antennas
- Each device has a single antenna
  - Can be extended to multi-antenna case
- Use Sliding BRNN detector to detect the received symbols at the BS without any knowledge of CSI



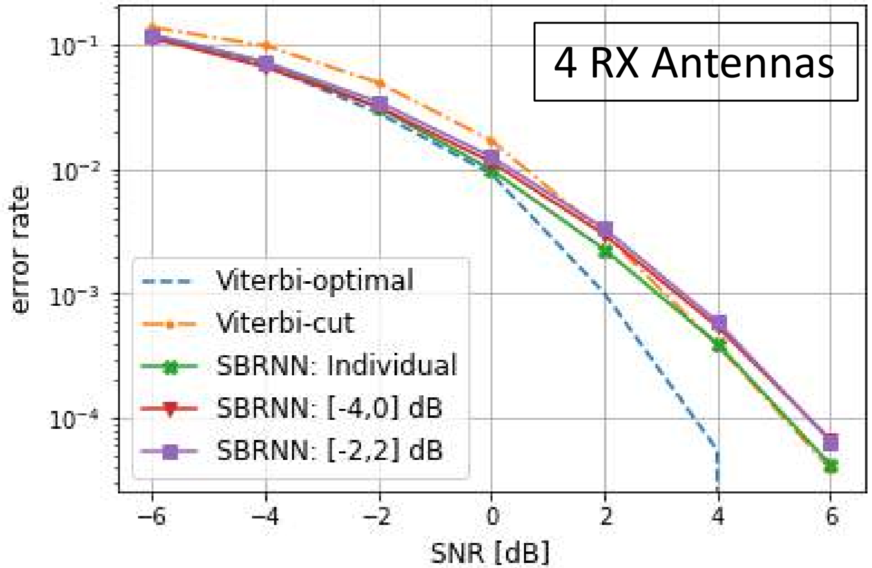


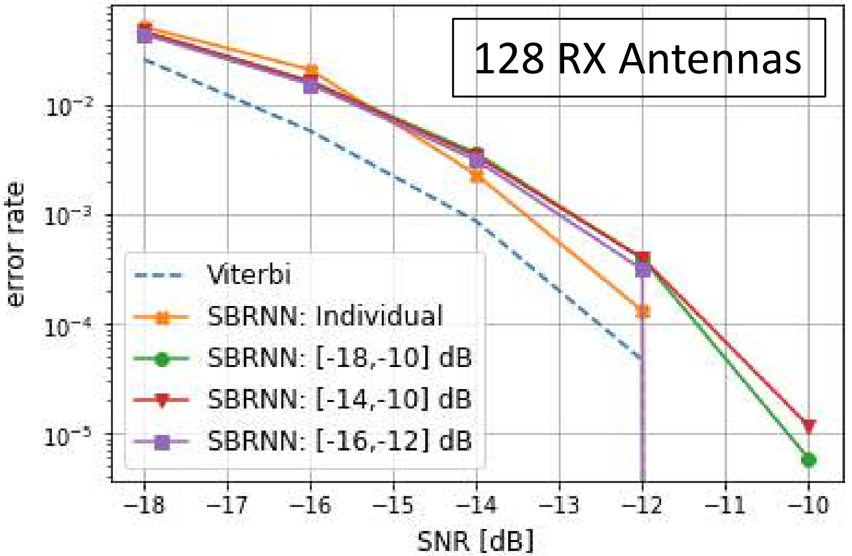


Performance: SBRNN vs Viterbi

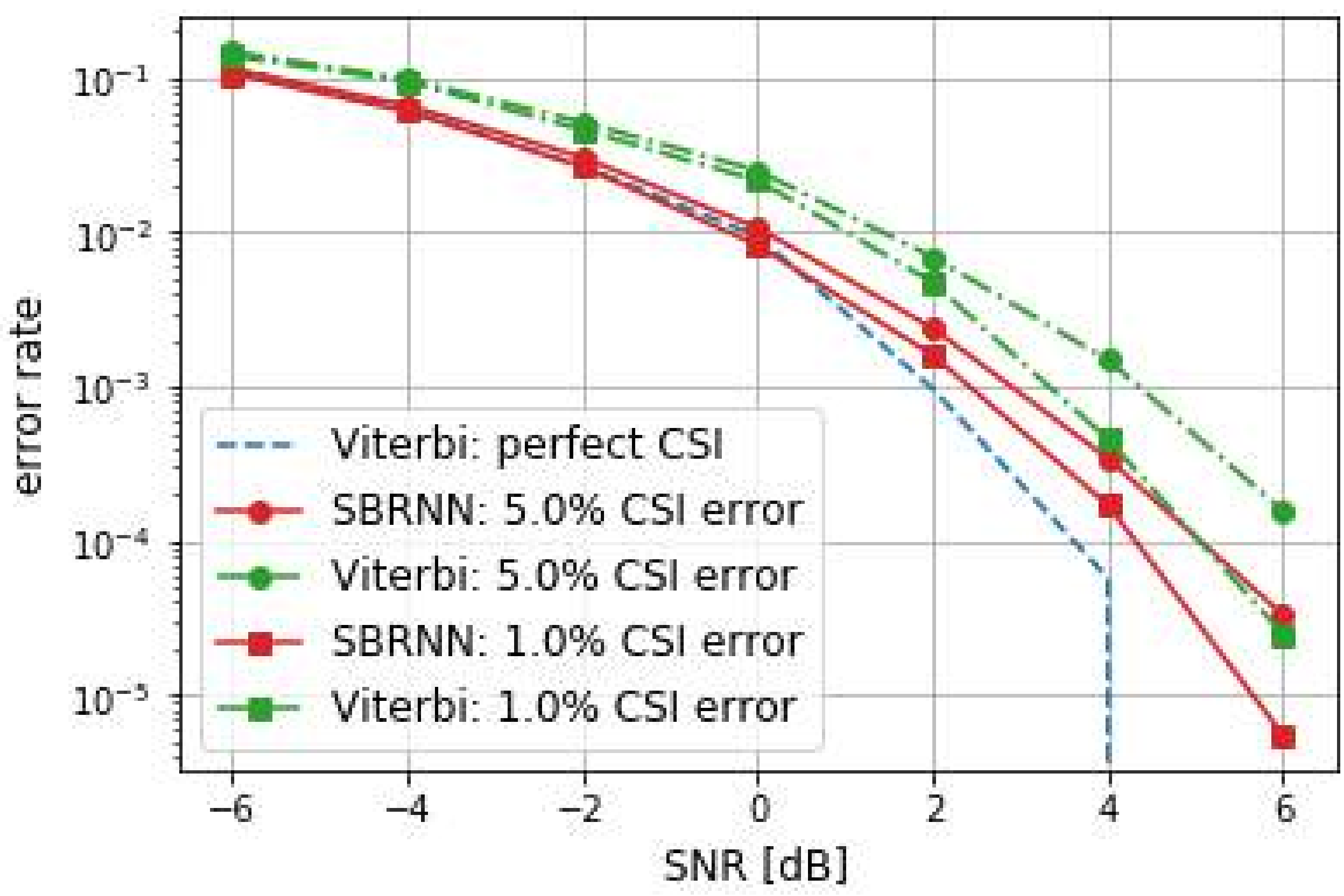
- SBRNN is close to the optimal Viterbi, and outperforms Viterbi-cut
- SBRNN trained on a single SNR range generalizes well to other SNRs.

Parameter	Value
Carrier Frequency	28 GHz
Bandwidth	800 MHz
Transmit Power	11 dBm
Tx-Rx Separation distance	60 m
Tx-Rx Antenna Gains	24.5 dBi
Modulation Scheme	BPSK

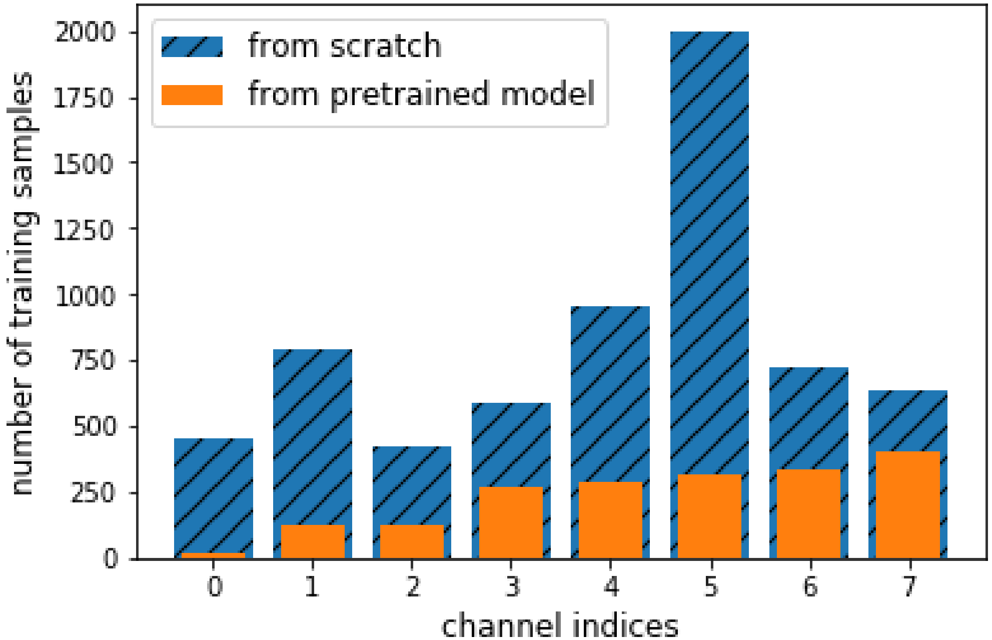




Performance: SBRNN w/ CSI imperfection



Training Overhead and Run Time Efficiency



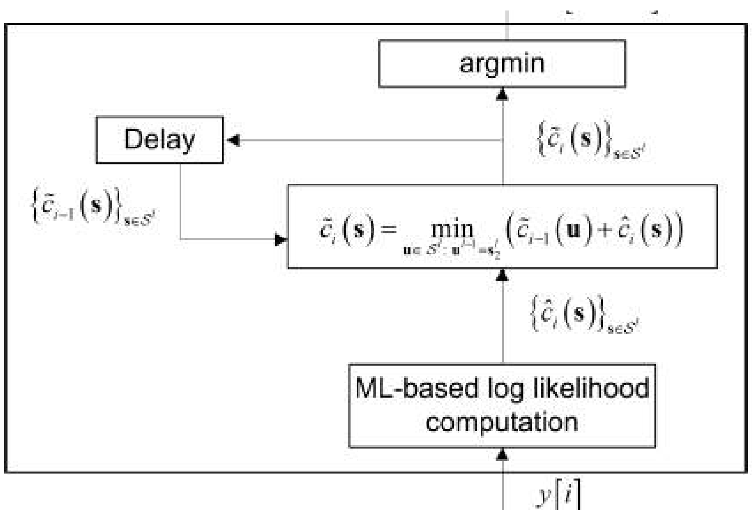
- Training: # of samples needed for 90% detection accuracy
- Testing: run time of detecting one 200-bit message[1]

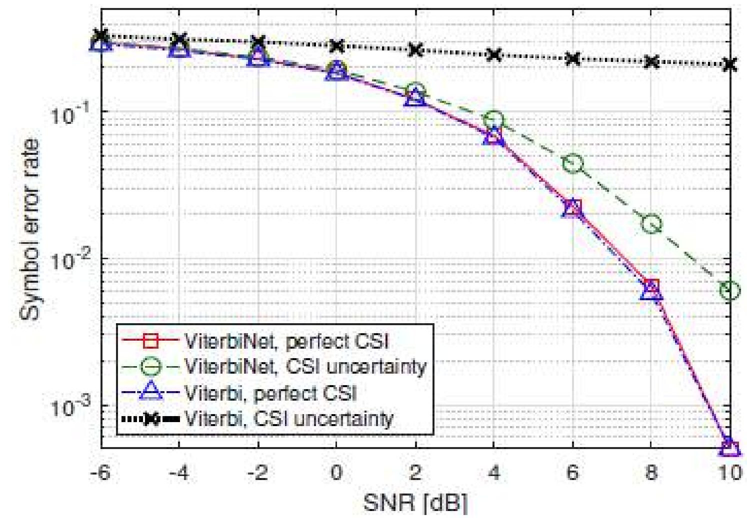
Receive Antennas	Sliding BRNN	Viterbi
4	0.244 s	12.461 s
128	0.264 s	52.681 s

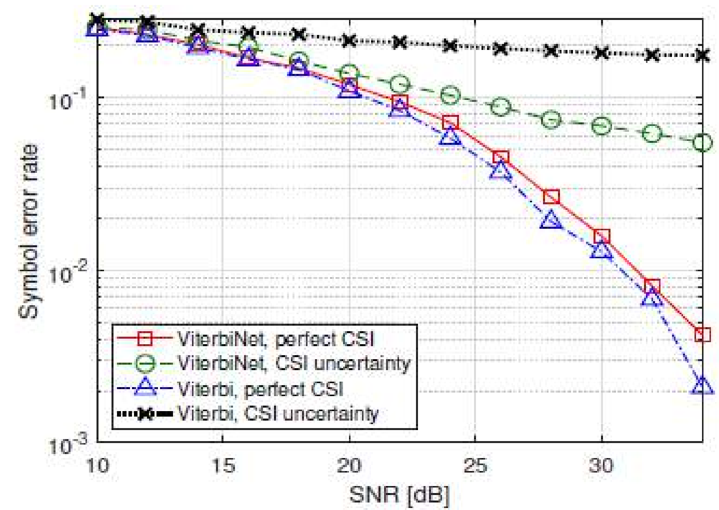
ViterbiNet: learn just p(y|x)

Algorithm 1 Viterbi Algorithm

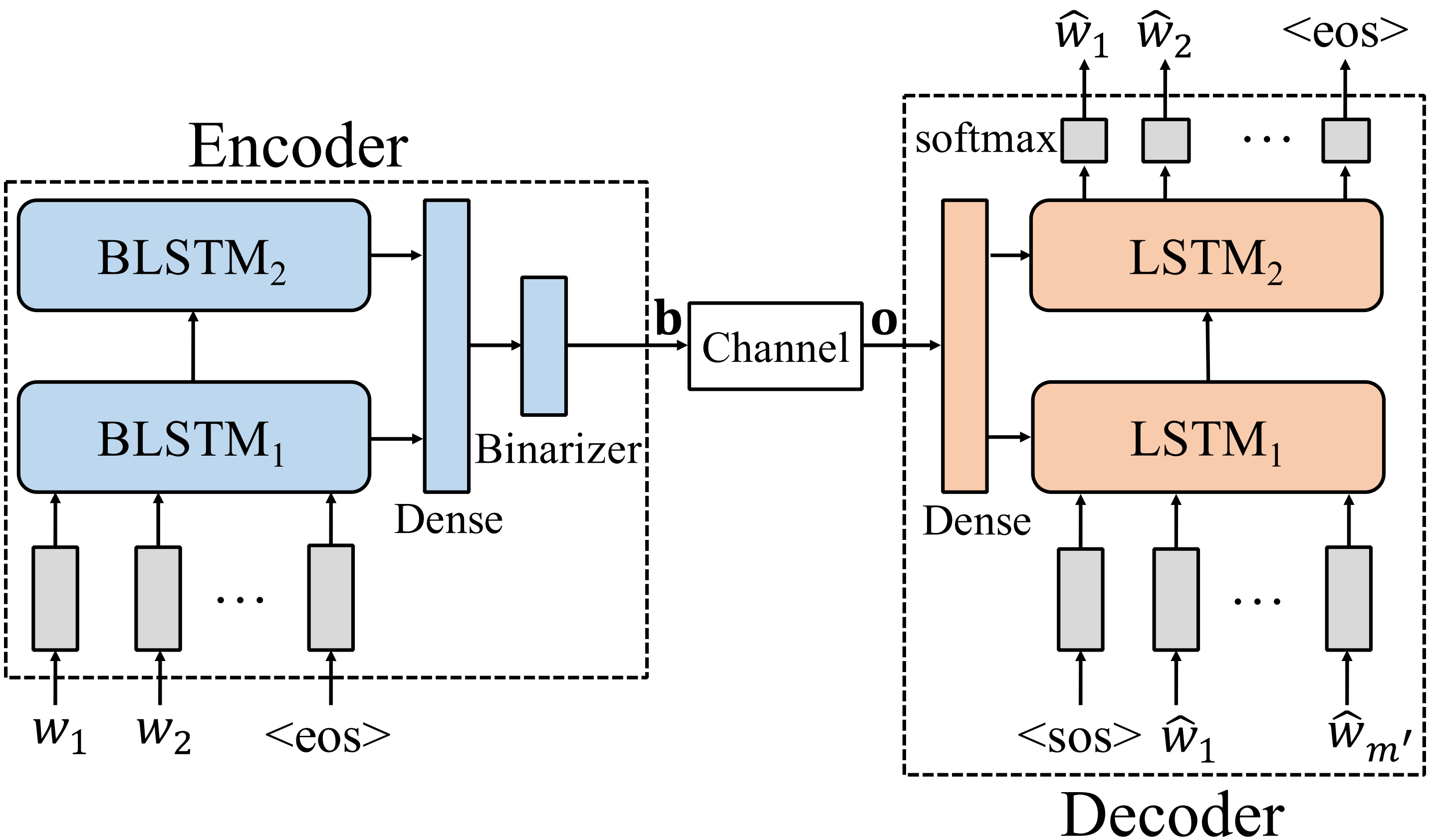
1: Input: Block of channel outputs $\boldsymbol{y}^t$, where $t > l$.
2: Initialization: Set $k = 1$, and fix $\tilde{c}_0(\tilde{s}) = 0$, $\forall \tilde{s} \in \mathcal{S}^l$.
3: Compute
$$\tilde{c}_k(\tilde{s}) = \min_{\boldsymbol{u} \in \mathcal{S}^l : \boldsymbol{u}^{l-1} = \tilde{\boldsymbol{s}}_2^l} \left(\tilde{c}_{k-1}(\boldsymbol{u}) + c_k(\tilde{s}) \right).$$
4: If $k \geq l$, set $(\hat{s})_{k-l+1} := (\tilde{s}^o)_1$, where
$$\tilde{s}^o = \arg\min_{\tilde{\boldsymbol{s}} \in \mathcal{S}^l} \tilde{c}_k(\tilde{s}).$$
5: Set $k := k + 1$. If $k \leq t$ go to Step 3.
6: Output: decoded output $\hat{s}^t$, where $\hat{s}^t_{t-l+1} := \tilde{s}^o$.







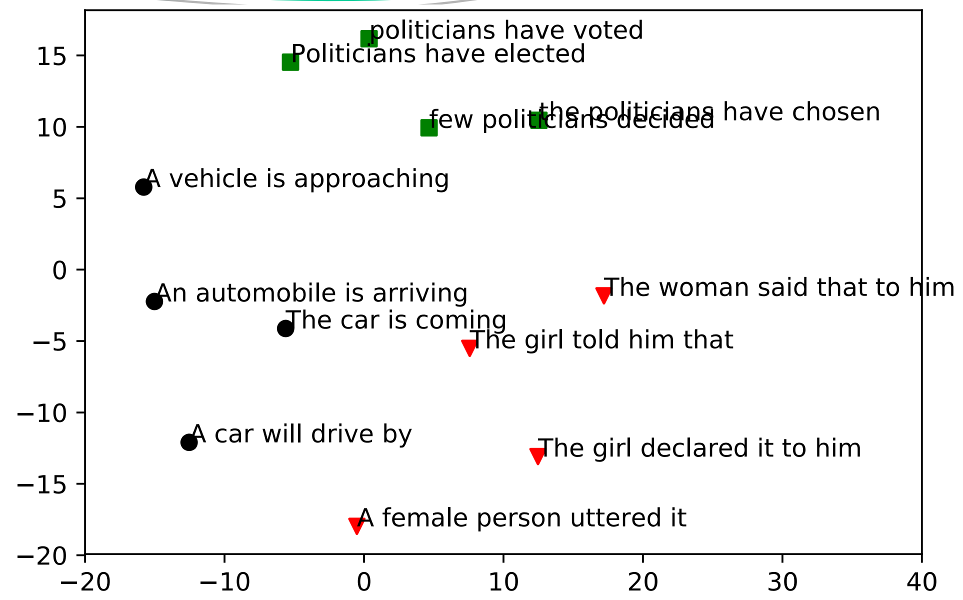
Joint Source Channel Coding Using Deep Learning

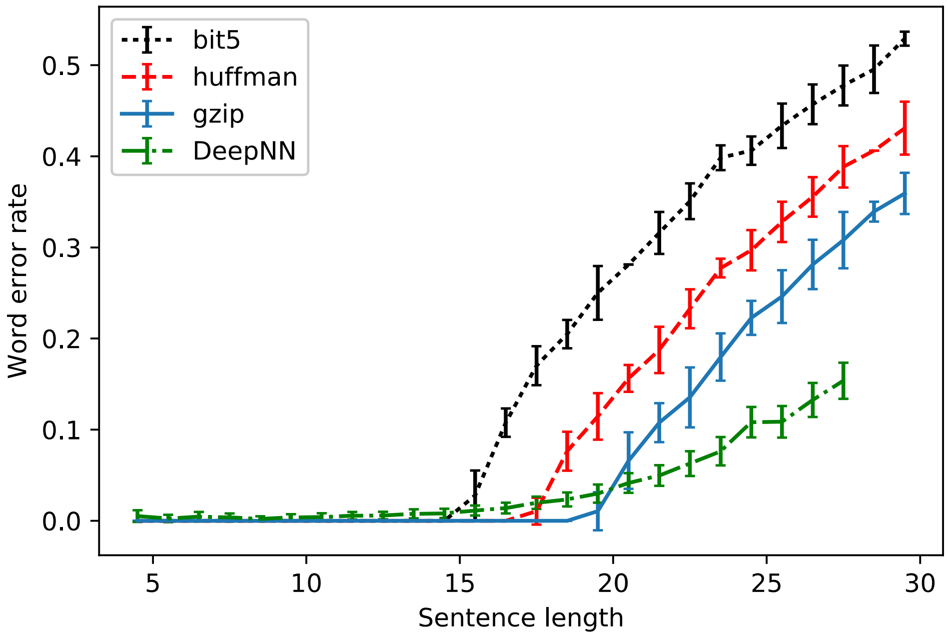


Results

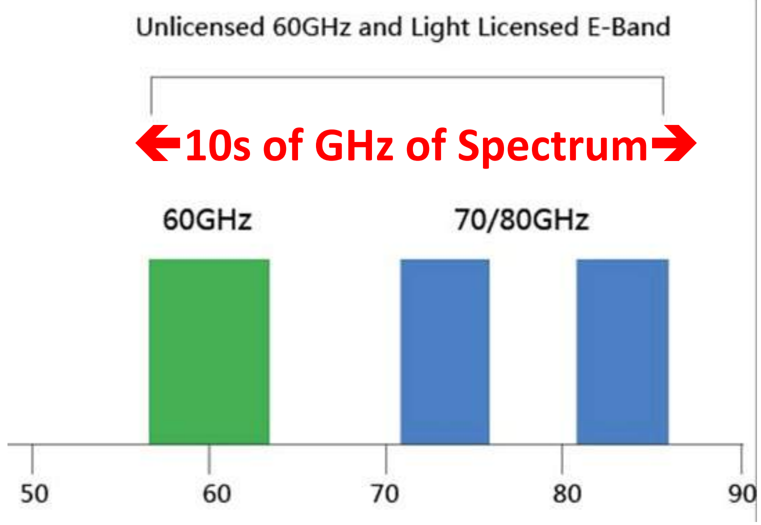
Top Plot:
- Sentences with similar meaning are placed close together in the code space

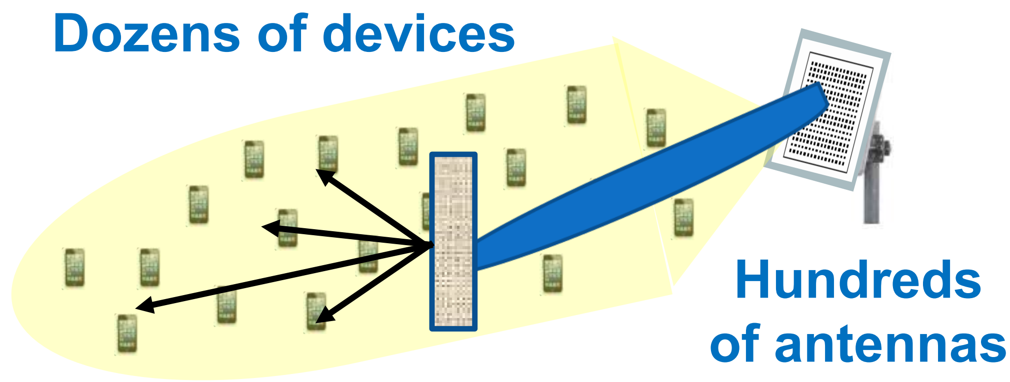
Bottom Plot:
- 400 bits per sentence,
- erasure rate of 0.05
- Reed-Solomon codes are used when separate source and channel code design is used (black, red, and blue)
- Considerable number of word errors for deep learning may be word replacements that preserve the meaning of the sentence





mmWave Massive MIMO





- mmWaves have large attenuation and path loss
- Massive MIMO removes attenuation, fading, interference
- Bottlenecks: channel estimation, complexity, propagation
- Ideal beamforming disappears with shadowing
- Need multihop/mesh networks

Blind MIMO Decoding via Vertex Hopping

Given samples in the form:

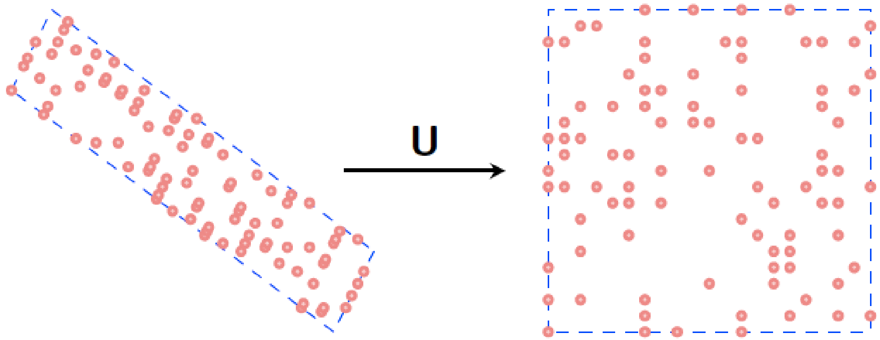
$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{e}$$

- $\boldsymbol{A} \sim \mathcal{N}(0,1)^{n\times n}, \boldsymbol{e} \sim \mathcal{N}(0,\sigma^2)^{n\times n}$.
- x is drawn from an MPAM or BPSK constellation (source must be hypercubic).
- Rich scattering, small MIMO ($2 \leq n \leq 12$).

Blind MIMO Decoding

In a block-fading environment, estimate A and recover $\boldsymbol{x}$ given only k samples of $\boldsymbol{y}$

Fitting a Parallelepiped --- Algorithms



Gaussian noise will not greatly distort this shape

$$\underset{\boldsymbol{U}}{\text{maximize}} \quad \log|\det \boldsymbol{U}| \qquad (*)$$

$$\text{subject to} \quad |\boldsymbol{U}\,\boldsymbol{y}_i|_\infty \le 1 + c, i = 1, \ldots, k$$

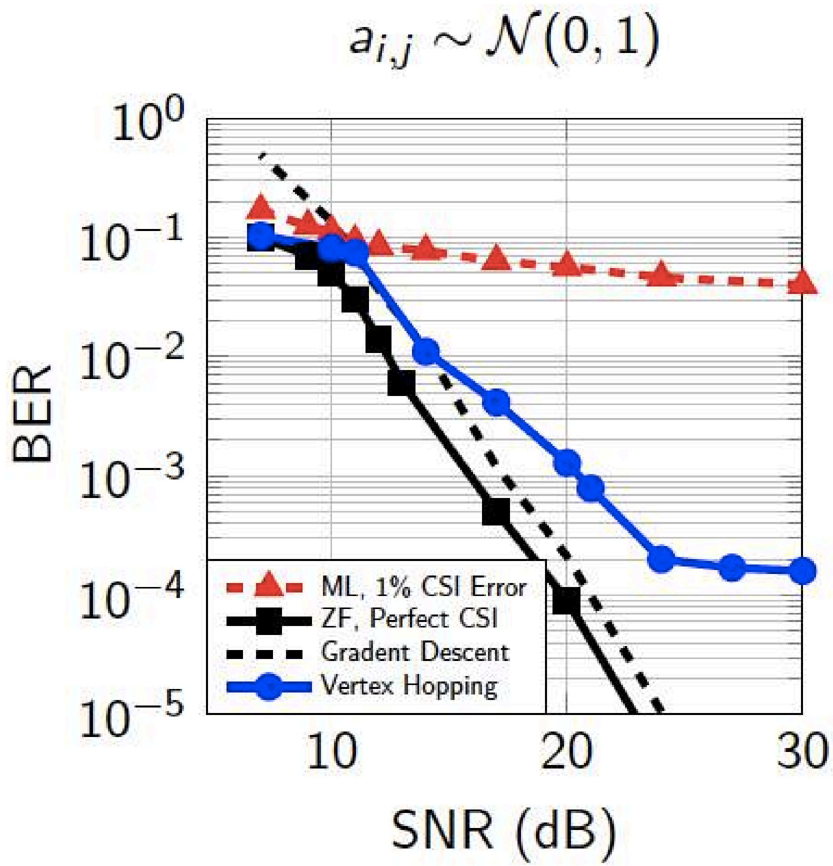
- $(*)$ is a non-convex optimization problem
- Constrained gradient descent works but is slow.
- The Vertex Hopping algorithm uses concepts from solving mixed-integer linear programming to solve $(*)$.

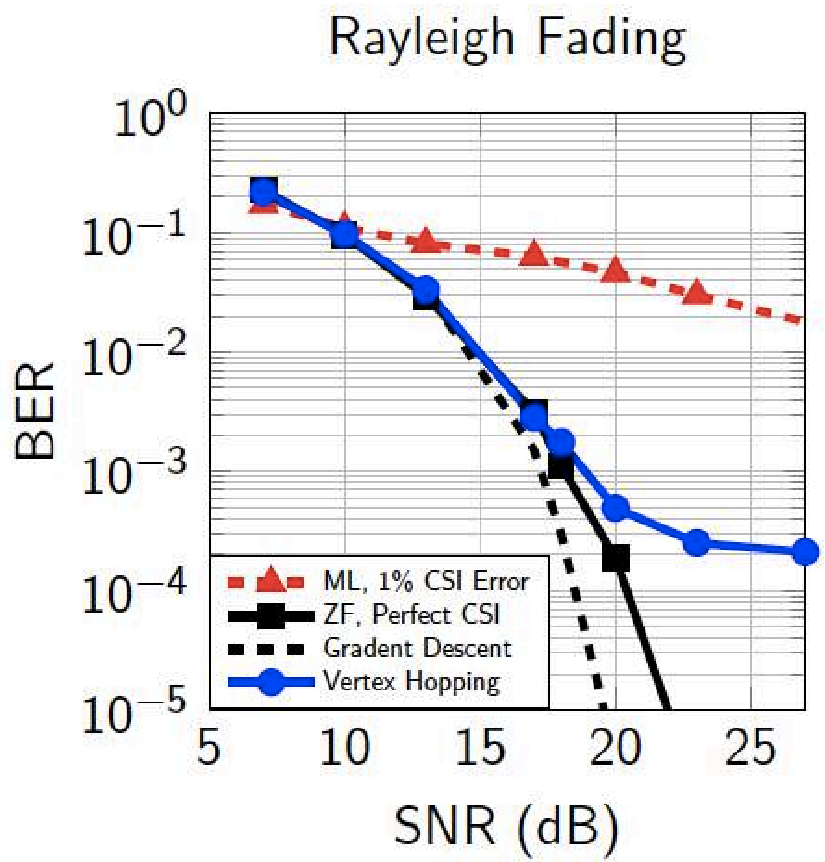
Runtime Performance

		Vertex Hopping*		Gradient Descent†	
n	k	Pr[success]	Time (s)	Pr[success]	Time (s)
2	8	1.0	1.83×10^{-5}	0.99	3.01×10^{-2}
3	13	1.0	6.46×10^{-5}	0.99	6.33×10^{-2}
4	18	1.0	1.74×10^{-4}	0.99	0.13
5	18	1.0	2.96×10^{-4}	0.97	0.30
6	22	1.0	8.52×10^{-4}	0.93	0.59
8	30	0.99	4.99×10^{-3}	0.80	3.5
10	45	0.99	5.36×10^{-2}	0	-
12	60	0.99	3.70×10^{-1}	0	-

***Implemented in Rust**
†MATLAB's `fmincon`

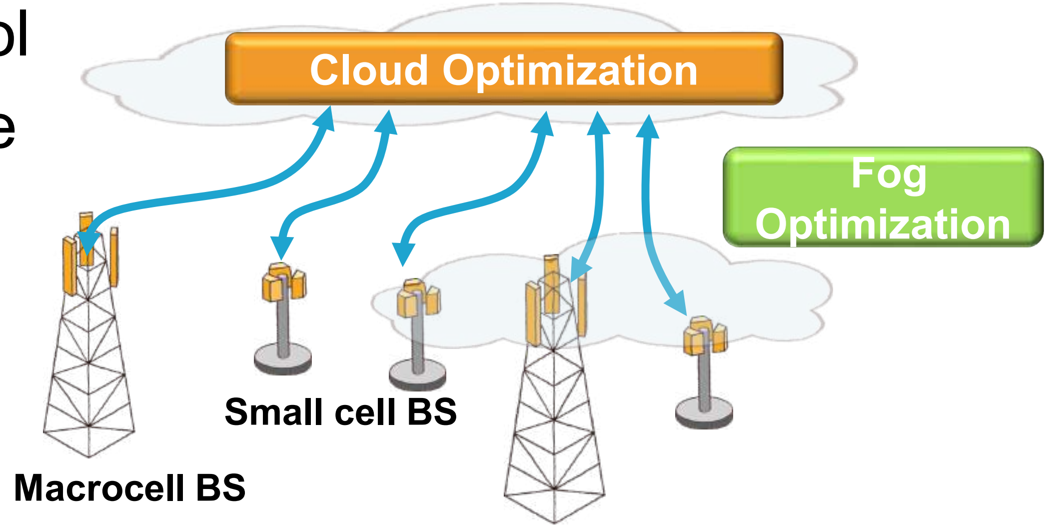
AWGN and Fading Performance





Network Optimization Challenges

- Algorithmic complexity
  - Frequency allocation alone is NP hard
  - Also have MIMO, power control, hierarchical networks: *NP-really-hard*
  - Advanced optimization tools needed, including a combination of centralized (cloud) distributed, and locally centralized (fog) control
  - ML can also play a role

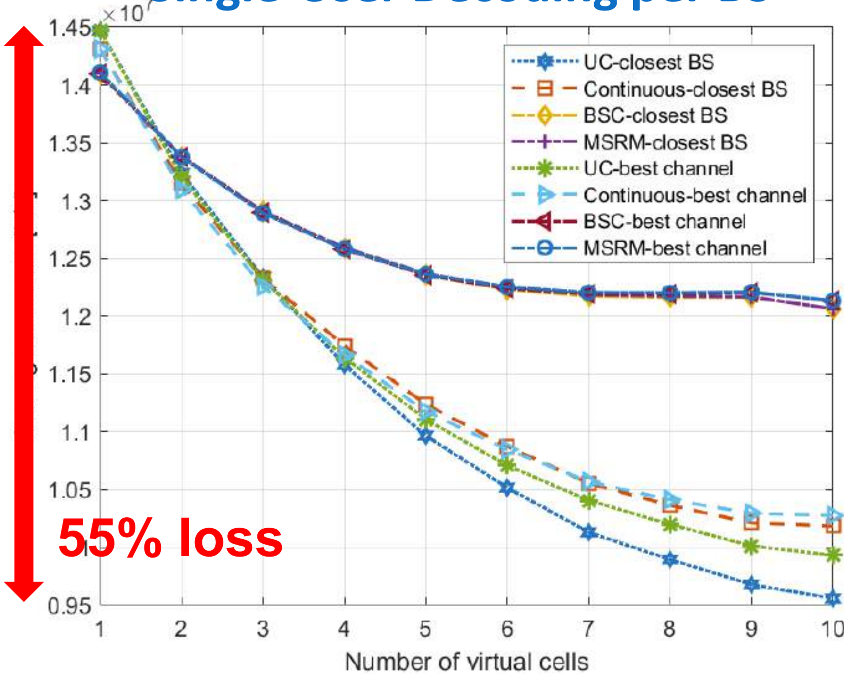


Next challenge: optimizing caching and edge computing

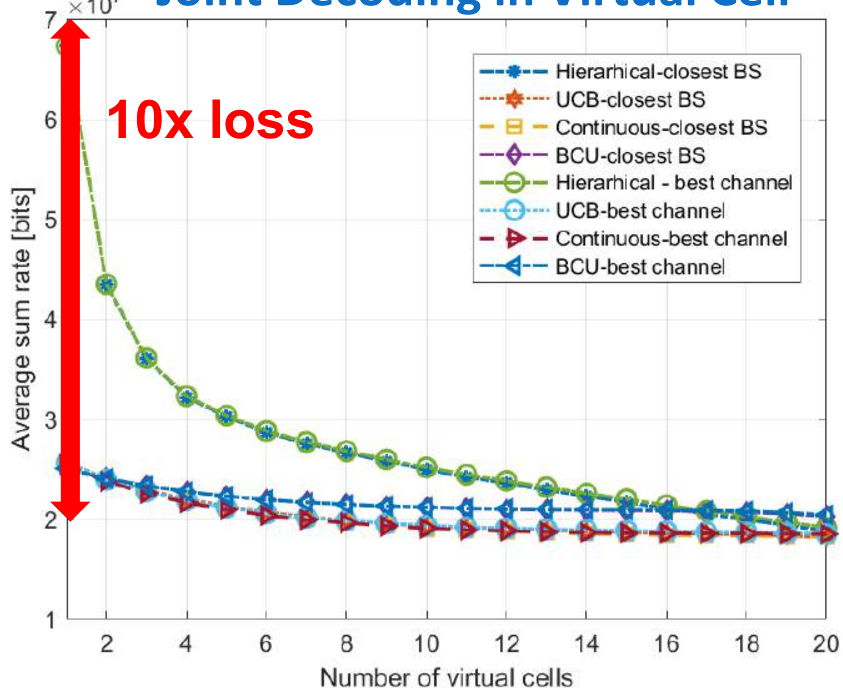
Fog-Optimization vs. Centralized

- Use clustering technique to cluster BSs, then optimize power allocation to maximize uplink sum rate
  - Consider multiple clustering techniques (not much difference)
  - Nonconvex approximation for optimization

Single-User Decoding per BS



Joint Decoding in Virtual Cell



Summary

- Future wireless networks must support high rates, extreme energy efficiency, and low latency
  - Small cells, multihop routing and massive MIMO are key enablers.
  - Network must be robust to rapidly-varying channels and adversaries
  - Machine learning and optimization is a promising new tool to use in receiver design, multiple access, and resource allocation
- Cloud and fog-based networking has many open challenges, particularly edge vs. cloud optimization