Spatial Deep Learning for Wireless Scheduling

Wei Yu

(Joint work with Wei Cui and Kaiming Shen)

Electrical and Computer Engineering Department
University of Toronto

2019
Machine learning is having tremendous successes in many tasks:
- Image classification; speech recognition/translation; face recognition...
- Superhuman performance is now commonplace.
Introduction

Machine vs. Human

- In many tasks, machines have *always* been better than human:
  - Examples: numerical calculation; circuit simulation; coding/decoding
  - Simple low-dimensional input – complex calculation
  - Accurate computational models are available.

- Surprise is that machines are now better in *uniquely human* tasks:
  - Complex high-dimensional input – making judgements
  - Accurate computational models are NOT available.
In communication engineering, channel models are cherished:
- Link level: Additive white Gaussian noise (AWGN) model is justified.
- Network level: Opportunities abound when there is human element.
- System level: Can machine learn to perform complex optimization?

Main point of this talk: The role of machine learning is when
- Models are expensive to obtain.
- Optimization is complex and difficult to perform.
Link Scheduling in Device-to-Device Networks
The data rate of link $i$ is limited by interference:

$$R_i(x) = \log \left( 1 + \frac{|h_{ii}|^2 p_i x_i}{\sum_{j \in \mathcal{L}, j \neq i} |h_{ij}|^2 p_j x_j + \sigma^2} \right).$$

Coordinated scheduling and power control:

- Scheduling: Select a subset of links to activate, i.e., $x_i \in \{0, 1\}$.
- For this talk, we assume fixed power $p_i$.

This is an NP-hard discrete optimization problem.

Its relaxation is non-convex: $x$ in both numerator & denominator.
Traditional Optimization Based Approach

- Formulate a weighted sum-rate maximization problem:

\[
\max \sum_{i \in \mathcal{L}} w_i \log \left( 1 + \frac{|h_{ii}|^2 p_i x_i}{\sum_{j \in \mathcal{L}, j \neq i} |h_{ij}|^2 p_j x_j + \sigma^2} \right).
\]

subject to \( x_i \in \{0, 1\} \)

- Traditional approach: Two-step process
  - Obtain channel state information (CSI): \( \theta = \{h_{ij}\} \).
  - Solve the optimization problem to obtain \( x^* \) given \( \theta \).

- This traditional approach faces two challenges:
  - Solving the optimization problem is hard due to non-convexity.
  - Obtaining CSI is expensive due to the limited coherence time.
Outline of This Talk

1. Part I: Fractional Programming
2. Part II: Learn to Optimize
Part I: Scheduling via Fractional Programming
Single-Ratio Fractional Programming (FP)

- Given functions $A(x) \geq 0$ and $B(x) > 0$, a *single-ratio* FP problem is
  \[
  \max_x \frac{A(x)}{B(x)}
  \]
  subject to \(x \in \mathcal{X}\).

- A classic approach is to decouple the ratio by Dinkelbach’s transform:
  \[
  \max_x A(x) - yB(x)
  \]
  subject to \(x \in \mathcal{X}\).

  Then update \(y = \frac{A(x)}{B(x)}\).

- When \(A(x)\) is concave and \(B(x)\) is convex, this leads to global optimal \(x\).

*However, this cannot be extended to the multiple-ratio case.*
A New Quadratic Transform

- We propose a novel quadratic transform that reformulates

\[
\underset{x}{\text{maximize}} \quad \frac{A(x)}{B(x)}
\]

subject to \(x \in \mathcal{X}\).

as the following problem:

\[
\underset{x,y}{\text{maximize}} \quad 2y \sqrt{A(x)} - y^2 B(x)
\]

subject to \(x \in \mathcal{X}\).

- Both the optimal variable \(x^*\) and optimal objective value are the same.

- Proof: For fixed \(x\), the optimal \(y = \frac{\sqrt{A(x)}}{B(x)}\).

This transform can be readily extended to a multiple-ratio case.
Given $K$ pairs of $A_i(x) \geq 0$ and $B_i(x) > 0$, a multiple-ratio problem is

$$\max_{x} \sum_{i=1}^{K} \frac{A_i(x)}{B_i(x)}$$

subject to $x \in \mathcal{X}$.

By the quadratic transform, the problem is reformulated as

$$\max_{x,y} \sum_{i=1}^{K} \left( 2y_i \sqrt{A_i(x)} - y_i^2 B_i(x) \right)$$

subject to $x \in \mathcal{X}$.
The data rate of link $i$ is

$$R_i(x) = \log \left( 1 + \frac{|h_{ii}|^2 p_i x_i}{\sum_{j \in \mathcal{L}, j \neq i} |h_{ij}|^2 p_j x_j + \sigma^2} \right).$$

Formulate maximum weighted sum rate problem as a multi-ratio problem:

$$\max_x \sum_{i \in \mathcal{L}} w_i R_i(x)$$

subject to $x_i \in \{0, 1\}, \forall i$. 

Wei Yu  (University of Toronto)  Deep Learning for Wireless Scheduling  2019  13 / 44
Consider the problem $\max_x \log \left(1 + \frac{A}{B}\right)$.

Rewrite as $\max_x \log (1 + \gamma)$ subject to $\gamma = \frac{A}{B}$.

Introduce the Lagrangian $L = \log (1 + \gamma) - \lambda \left(\gamma - \frac{A}{B}\right)$.

... 

Reformulate $\max_x \log \left(1 + \frac{A}{B}\right) \iff \max_{(x,\gamma)} f_r(x, \gamma)$, where 

$$f_r(x, \gamma) = \log(1 + \gamma) - \gamma + \frac{(1 + \gamma)A}{A + B},$$
Reformulations for D2D System

- Applying this technique to D2D gives (recall $x_i \in \{0, 1\}$ is for schedule)

$$f_r(x, \gamma) = \sum_{i \in \mathcal{L}} w_i \log(1 + \gamma_i) - \sum_{i \in \mathcal{L}} w_i \gamma_i + \sum_{i \in \mathcal{L}} \frac{w_i (1 + \gamma_i)|h_{ii}|^2 p_i x_i}{\sum_{j \in \mathcal{L}} |h_{ij}|^2 p_j x_j + \sigma^2}.$$  

- Thus, $\max_x \sum w_i \log(1 + \text{SINR}_i) \iff \max_x, \gamma f_r(x, \gamma)$.

- For fixed $x$, the optimal $\gamma$ is (by solving $\partial f_r / \partial \gamma_i = 0$)

$$\gamma_i^* = \frac{|h_{ii}|^2 p_i x_i}{\sum_{j \in \mathcal{L}, j \neq i} |h_{ij}|^2 p_j x_j + \sigma^2}.$$
Further applying the quadratic transform to the last term of $f_r$ gives

\[
f_q(x, \gamma, y) = \sum_{i \in \mathcal{L}} 2y_i \sqrt{w_i(1 + \gamma_i)} |h_{ii}|^2 p_i x_i - \sum_{i \in \mathcal{L}} y_i^2 \left( \sum_{j \in \mathcal{L}} |h_{ij}|^2 p_j x_j + \sigma^2 \right) + \text{const}(\gamma).
\]

Thus, $\max_x \sum w_i \log(1 + \text{SINR}_i) \iff \max_x, \gamma f_r \iff \max_x, \gamma, y f_q$.

For fixed $x$ and $\gamma$, the optimal $y$ is (by solving $\partial f_q / \partial y_i = 0$)

\[
y_i^* = \frac{\sqrt{w_i(1 + \gamma_i)} |h_{ii}|^2 p_i x_i}{\sum_{j \in \mathcal{L}} |h_{ij}|^2 p_j x_j + \sigma^2}.
\]
Optimization of Scheduling Variable

- Note that $f_q$ can be decoupled on a per-link basis with respect to $x$:
  \[
  f_q(x, \gamma, y) = \text{const}(\gamma, y) + \sum_{i \in L} Q_i(x_i, \gamma, y)
  \]
  where the per-link function $Q_i$ is defined to be
  \[
  Q_i(x_i, \gamma, y) = 2y_i \sqrt{w_i(1 + \gamma_i)|h_{ii}|^2 p_i x_i} - \sum_{j \in L} y_j^2 |h_{ji}|^2 p_j x_j.
  \]
- The optimal solution for $x$ now becomes straightforward:
  \[
  x_i^* = \arg \max_{x_i} Q_i(x_i, \gamma, y).
  \]
Proposed FPLinQ for D2D Scheduling

Algorithm 1 FPLinQ for scheduling D2D links

0) Initialize all the variables to feasible values.

repeat

1) Update $\gamma$

2) Update $y$

3) Update $\tilde{x}$ (relax $x$ to real number);

until Convergence

4) Recover the integer $x$.

Here, $\gamma$ and $y$ are intermediate variables that coordinate the scheduling of the links and slow down the convergence, thus outperforming greedy.
Learn to Optimize?

- FPLinQ is highly effective, but it still requires CSI.
- Obtaining CSI is expensive due to limited coherence time.
- Can a machine learn the optimal solution directly?
Part II: Learn to Schedule without CSI
Learn to Optimize

- If optimization is hard, can we learn the optimal solution directly?

\[ \theta \rightarrow \max_x f(x; \theta) \rightarrow x^* \]

Instead of optimizing \( x \) for \( \theta = \{h_{ij}\} \), we learn the mapping \( \theta \rightarrow x^* \).

- Train a deep neural network to learn the functional mapping
  - **Supervised learning**: Using many examples of \((\theta, x^*)\) from FPLinQ.
  - **Unsupervised learning**: Directly maximizing \( f(x; \theta) \). Better strategy.

- Ask the deep neural network to produce the optimal \( x^* \) for a new \( \theta \).
Learn from Geographic Information

- For wireless scheduling, coherence time is limited, feedback is costly:
- Obtaining and feeding back $\theta = \{h_{ij}\}$ become the bottleneck:
  $$|\{h_{ij}\}| = O(N^2)$$
- Instead of the channels, we use geographic location information as $\theta$. Location information scales as $O(N)$.

$$\theta \rightarrow_{\max_x f(x; \theta)} x^*$$

Neural network learns to map the geographic information to $x^*$. 
Spatial Deep Learning Approach

- Randomly generate D2D network (e.g. 50 links over 500mx500m area)
- The network is represented Geographic Location Information (GLI): a set of vectors \( \{(l_{tx}^i, l_{rx}^i)\}_i \), where \( l_{tx}^i \in \mathbb{R}^2 \) and \( l_{rx}^i \in \mathbb{R}^2 \) are the transmitter and the receiver locations of the \( i \)th link, respectively.
- **Convolutional** neural network with geographic information as input.
  - Back propagation over convolutional filter and connection weights.
  - **Unsupervised learning** with sum-rate as the objective function.
  - Testing/Validation on new D2D networks.

Novel Deep Neural Network

The overall network structure consists of multiple feedback stages:

- Each iteration has a convolution and a fully connected forward path.
**Density Grid**

- Two density grid matrices are constructed with $5m \times 5m$ cells to represent the density of the active transmitters and receivers:

$$T(s,t) = \sum_{\{i| (s_{tx}^i, t_{tx}^i) = (s,t)\}} x_i$$

$$R(s,t) = \sum_{\{i| (s_{rx}^i, t_{rx}^i) = (s,t)\}} x_i.$$
Learn to Optimize Spatial Learning

Forward Path of Each Link

Filter Center Anchor: Receiver’s location

Filter Center Anchor: Transmitter’s location

Convolution Filter

ReLu nonlinearity

Sigmoid nonlinearity

DCS*: Direct Channel Strength

*DCS: Direct Channel Strength

Feature Vector Per Link

Output Per Link

Transmitter Density Grid

Receiver Density Grid

Previous Iteration Allocation ($0 \sim 1$)
**Convolution Filter: Summarizing the Interference**

- The convolution filter range is up to 315 meters \( \times \) 315 meters.
- Each link’s own transmitter/receiver is subtracted from convolution.

- Radial pattern of the filter indicates delaying interference intensity.
Channel strength is estimated by extracting weight of trained filter.
Fully Connected Stage

- After convolution stage, we form the **feature vector** for each link:
  - The total interference the transmitter causes to other links
  - The total interference the receiver is subject to by other links
  - The link strength and its range over the layout
  - The allocation status of the link from previous iteration via feedback

- This feature vector serves as input for the fully connected stage, with standard fully connected hidden layers with ReLU nonlinearities.

- The last layer uses sigmoid nonlinearity to squash the outputs into $[0, 1]$, indicating the power allocation of the link at end of iteration.
Feedback Connection

- The overall network structure consists of multiple feedback stages:

- We use $x_i$ from the previous iteration to update the density grids, as well as forming the feature vector of the current iteration.
Stochastic Updates

- Oscillating ON/OFF behavior may occur

- Randomization is performed to break these oscillations.
Simulation Model

- Full frequency reuse with 5MHz bandwidth at 2.4GHz carrier frequency; 1.5m antenna height and 2.5dB antenna gain.
- Additive white Gaussian noise at -169dBm/Hz
- SNR gap at 6dB
- Max transmit power is set to be constant across each link at 40dBm
- Short-range outdoor model ITU-1411 distance-dependent pathloss
Training Configurations

- Each layout consists of 50 randomly placed D2D links over 500 meters × 500 meters region.
- Neural network is trained with datasets of 800,000 layouts with either
  - Tx-to-Rx distance uniformly distributed in 2 meters ∼ 65 meters, or
  - Tx-to-Rx distance uniformly distributed in 30 meters ∼ 70 meters.
- In the training stage, the feedback runs for 5 iterations.
- In the testing stage, the feedback runs for 30 iterations.
Sum Rate Maximization: Same Training/Testing Setting

Table: Sum rate of 50 links over 500m×500m area, with 30m ~ 70m Tx-to-Rx distance distribution, over 5000 testing layouts

<table>
<thead>
<tr>
<th></th>
<th>CSI</th>
<th>No Fading</th>
<th>With Fading</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spatial Deep Learning</td>
<td>–</td>
<td>92.2</td>
<td>71.8</td>
</tr>
<tr>
<td>Greedy</td>
<td>✓</td>
<td>84.8</td>
<td>95.9</td>
</tr>
<tr>
<td>Strongest Links</td>
<td>✓</td>
<td>59.7</td>
<td>65.4</td>
</tr>
<tr>
<td>Random Selection</td>
<td>–</td>
<td>35.3</td>
<td>31.7</td>
</tr>
<tr>
<td>All Active</td>
<td>–</td>
<td>26.7</td>
<td>25.3</td>
</tr>
<tr>
<td>FP</td>
<td>✓</td>
<td>100</td>
<td>100</td>
</tr>
</tbody>
</table>

If scheduling using FP without fast fading, FP achieves 77.7%.
Generalizability

- We vary the network layouts to test the robustness of our approach:
  - Larger area with more links while maintaining link density
  - Same area but with more links and higher link densities
  - Layouts with varying distributions for Tx-to-Rx distances
## Sum Rate Maximization: Generalizability

The table below shows the sum rate simulation results for different Tx-to-Rx distance distributions and network sizes.

### Table: 30m ~ 70m Tx-to-Rx distance distribution

<table>
<thead>
<tr>
<th>Sum Rate (%)</th>
<th>FP</th>
<th>Neural Network</th>
<th>Greedy</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 Links; 1000×1000 m²</td>
<td>100</td>
<td>94.58</td>
<td>104.68</td>
</tr>
<tr>
<td>450 Links; 1500×1500 m²</td>
<td>100</td>
<td>95.46</td>
<td>106.03</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sum Rate (%)</th>
<th>FP</th>
<th>Neural Network</th>
<th>Greedy</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 Links; 500×500 m²</td>
<td>100</td>
<td>92.17</td>
<td>89.73</td>
</tr>
<tr>
<td>500 Links; 500×500 m²</td>
<td>100</td>
<td>91.35</td>
<td>92.41</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sum Rate (%)</th>
<th>FP</th>
<th>Neural Network</th>
<th>Greedy</th>
</tr>
</thead>
<tbody>
<tr>
<td>10m ~ 50m</td>
<td>100</td>
<td>98.44</td>
<td>94.00</td>
</tr>
<tr>
<td>30m ~ 100m</td>
<td>100</td>
<td>88.12</td>
<td>85.51</td>
</tr>
<tr>
<td>30m fixed</td>
<td>100</td>
<td>96.64</td>
<td>84.56</td>
</tr>
</tbody>
</table>
Complexity of neural network is $O(N)$. Greedy and FP are both $O(N^2)$. 
Proportional Fairness Scheduling

- Consider long-term average rate over exponentially weighted window:

\[
\bar{R}_i^t = (1 - \alpha) \bar{R}_i^t + \alpha R_i^t
\]

- Proportional fairness scheduling aims to optimize:

\[
\sum_{i=1}^{N} \log(\bar{R}_i).
\]

- Equivalently, we optimize weighted sum rate:

\[
\sum_{i=1}^{N} w_i R_i^t
\]

where

\[
w_i = \left. \frac{\partial U(\bar{R}_i^t)}{\partial R} \right|_{\bar{R}_i^t} = \left. \frac{\partial \log(\bar{R}_i^t)}{\partial R} \right|_{\bar{R}_i^t} = \frac{1}{\bar{R}_i^t}.
\]
**New Idea:** The weighted sum rate optimization at each time slot could be approximated by sum rate optimization on a subset of links.

We aim to use binary weights to approximate real weights:

- Let $w^t$ denote the original proportional fairness weight vector.
- Find a binary vector $\hat{w}^t$ to minimize the angle in between.

\[ \theta \]

\[ \frac{w^t}{\|w^t\|_2} \]

\[ \frac{\hat{w}^t}{\|\hat{w}^t\|_2} \]

- The “best” binary approximation to the original real weight vector.
### Proportionally Fair Scheduling

**Table:** Sum rate of 50 links over 500m × 500m area, with 30m ~ 70m Tx-to-Rx distance distribution, for 10 testing layouts over 500 scheduling slots

<table>
<thead>
<tr>
<th>Methods</th>
<th>CSI</th>
<th>Sum Log Utility</th>
<th>5-Percentile Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spatial Deep Learning</td>
<td>-</td>
<td>45.35</td>
<td>1.40 Mbps</td>
</tr>
<tr>
<td>Greedy</td>
<td>✓</td>
<td>39.63</td>
<td>1.88 Mbps</td>
</tr>
<tr>
<td>Random Selection</td>
<td>-</td>
<td>0.90</td>
<td>0.33 Mbps</td>
</tr>
<tr>
<td>All Active</td>
<td>-</td>
<td>-27.59</td>
<td>0.07 Mbps</td>
</tr>
<tr>
<td>FP</td>
<td>✓</td>
<td>45.24</td>
<td>1.35 Mbps</td>
</tr>
</tbody>
</table>
Weighted Sum Rate Maximization

Cumulative Distribution of Average Rates over the Links

Mean Rate for each link (Mbps)

Cumulative Distribution Function

Deep Learning
FP
Weighted Greedy
Max Weight
All Active
Random

Wei Yu (University of Toronto)
Deep Learning for Wireless Scheduling
2019 41 / 44
Summary

- We propose a new fractional programming method for link scheduling.
- We propose a novel neural network for scheduling interfering D2D links, bypassing channel estimation, while achieving good performance.
- Key features:
  - Unsupervised learning using the sum rate as the optimization objective
  - Spatial convolution of geographic information to estimate interference
  - Per-link fully connected neural network to make scheduling decision
  - Overall feedback structure with stochastic update
Concluding Remarks

- Traditional communication system design: Model then optimize.

- Machine learning approach:
  - Use a universal and highly expressive model, e.g. deep neural network
  - Rely on large amount of training data.

- Machine learning is most useful when:
  - Models are difficult or expensive to obtain.
  - Inputs are high-dimensional or heterogeneous.
  - Computational complexity of producing optimized output is high.

- Matching neural network architecture to problem structure is key.
Further Information

Wei Cui, Kaiming Shen, and Wei Yu,
“Spatial Deep Learning for Wireless Scheduling”,
*IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1248-1261, June 2019.

Kaiming Shen and Wei Yu,
“Fractional Programming for Communication Systems – Part I: Power Control and Beamforming”,

Kaiming Shen, and Wei Yu,
“Fractional Programming for Communication Systems – Part II: Uplink Scheduling via Matching”,