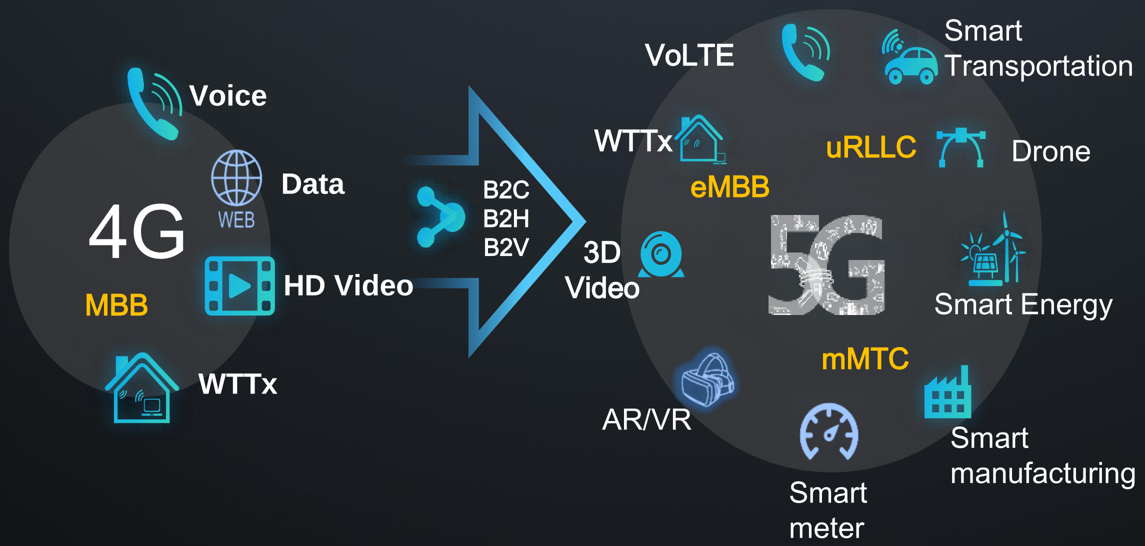
Mobile AI: Challenges and Opportunities

Mérouane Debbah

Mathematical and Algorithmic Sciences Lab, Huawei

Geneva, January 29th, 2018

Networks are becoming very complex



One Network 4 Services

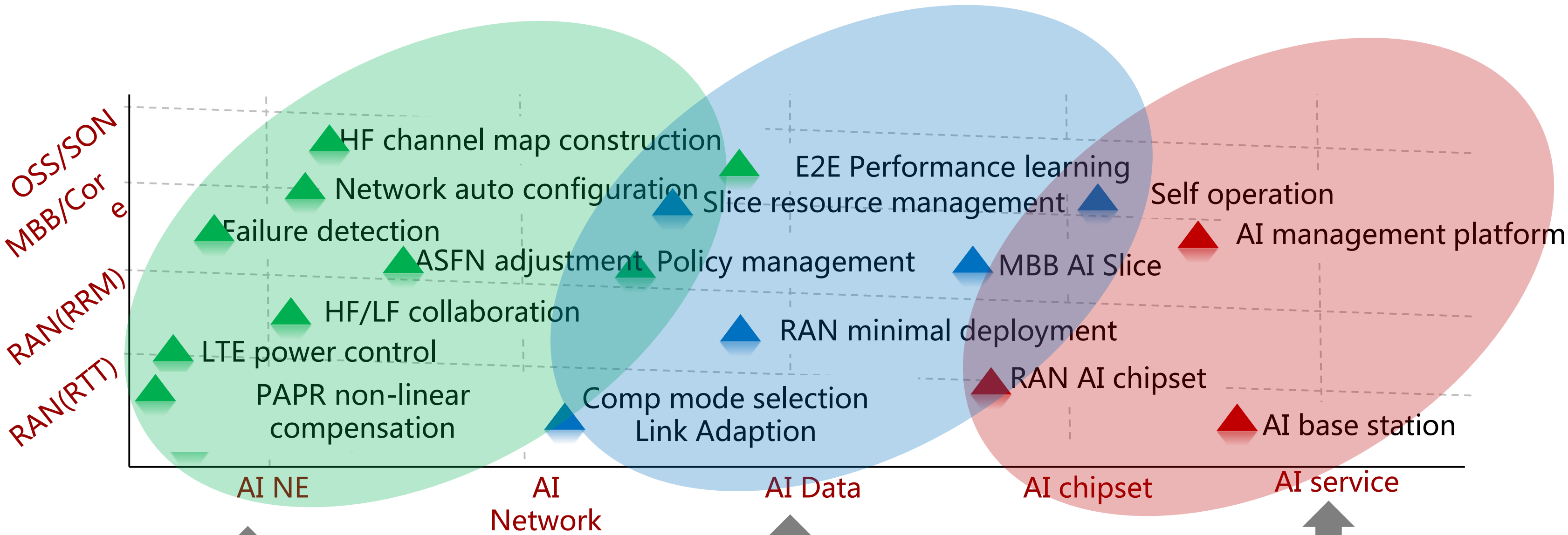
One Network Hundreds of Services

Wireless AI

Big data and machine learning technology network management can overcome the Wireless SON network management problem when the network cannot be accurately expressed with formulas.

	AI Algorithm	Wireless Algorithm
Value	Data	Link
Scenario	Automatically	Manually
Target	Global probability optimization	Local determined optimization
Scope	E2E network	Locally Modelling
method	Big data, learning	Formula , optimization
Usage	Set the target goal	Tune parameters manually

AI in Wireless



Algorithm

- Regression
- Clustering
- Classification

- Re-enforcement learning
- Dynamic optimization
- GMM/HMM
- Association rule mining

- Deep Learning
- RDL
- Transfer Learning
- Graphic algorithms

AI in Telecommunications is not new

Philosophical Magazine, Ser.7, Vol. 41, No. 314 - March 1950.

XXII. Programming a Computer for Playing Chess[1]

By CLAUDE E. SHANNON

Bell Telephone Laboratories, Inc., Murray Hill, N.J.[2]

[Received November 8, 1949]

1. INTRODUCTION

This paper is concerned with the problem of constructing a computing routine or "program" for a modern general purpose computer which will enable it to play chess. Although perhaps of no practical importance, the question is of theoretical interest, and it is hoped that a satisfactory solution of this problem will act as a wedge in attacking other problems of a similar nature and of greater significance. Some possibilities in this direction are: -

AI in Telecommunications is not new

(1)Machines for designing filters, equalizers, etc.
(2)Machines for designing relay and switching circuits.
(3)Machines which will handle routing of telephone calls based on the individual circumstances rather than by fixed patterns.
(4)Machines for performing symbolic (non-numerical) mathematical operations.
(5)Machines capable of translating from one language to another.
(6)Machines for making strategic decisions in simplified military operations.
(7)Machines capable of orchestrating a melody.
(8)Machines capable of logical deduction.

AI in Telecommunication is not new

Unfortunately a machine operating according to the type A strategy would be both slow and a weak player. It would be slow since even if each position were evaluated in one microsecond (very optimistic) there are about 10^9 evaluations to be made after three moves (for each side). Thus, more than 16 minutes would be required for a move, or 10 hours for its half of a 40-move game.

It would be weak in playing skill because it is only seeing three moves deep and because we have not included any condition about quiescent positions for evaluation. The machine is operating in an extremely inefficient fashion - it computes all variations to exactly three moves and then stops (even though it or the opponent be in check). A good human player examines only a few selected variations and carries these out to a reasonable stopping point. A world champion can construct (at best) combinations say, 15 or 20 moves deep.

Brain Empowered Wireless Communications

IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, VOL. 23, NO. 2, FEBRUARY 2005 201

Cognitive Radio: Brain-Empowered Wireless Communications

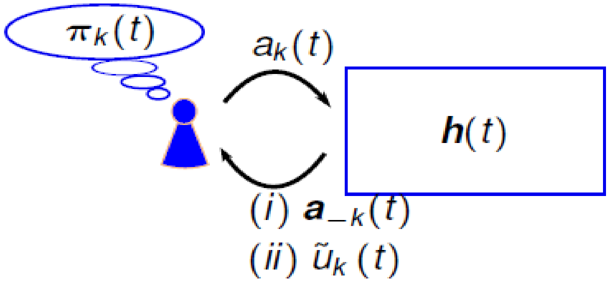
Simon Haykin, *Life Fellow, IEEE*

Invited Paper

Learning Algorithms for SON

Learning Iterative Steps:

- **Choose** action $a_k(t) \sim \pi_k(t)$.
- **Observe** game outcome, e.g.,
  $\boldsymbol{a}_{-k}(t)$
  $u_k(a_k(t), \boldsymbol{a}_{-k}(t))$.
- **Improve** $\pi_k(t+1)$.



Thus, we can expect that: $\forall k \in \mathcal{K}$,

$$\pi_k(t) \xrightarrow{t \to \infty} \pi_k^* \qquad (1)$$

$$\bar{u}_k(\pi_k(t), \pi_{-k}(t)) \xrightarrow{t \to \infty} \bar{u}_k(\pi_k^*, \pi_{-k}^*) \qquad (2)$$

where, $\pi^* = (\pi_1^*, \ldots, \pi_K^*)$ is a NE strategy profile.

Learning Algorithms for SON

- Best Response Dynamics (BRD)
- Fictitious Play (FP)
- Reinforcement Learning (RL)
- Joint Utility Strategy Learning (JUSTE)
- Trial and Error Learning (TE)
- Regret Matching Learning
- Q-Learning
- Multi-Arm Bandits
- Imitation Learning

The exploration versus exploitation dilemma

	BRD	RL	JUSTE	TE
Convergence	NE	$--$	ϵ-NE	PNE
Observations	$\boldsymbol{a}_{-k}(t)$	$\tilde{u}_k(t)$	$\tilde{u}_k(t)$	$\tilde{u}_k(t)$
Closed Expr. for u_k	Yes	No	No	No
Calculation. Cap.	Optim.	Algeb. Oper.	Algeb. Oper.	Algeb. Oper.
Environment	Static	Stationary	Stationary	Dynamic

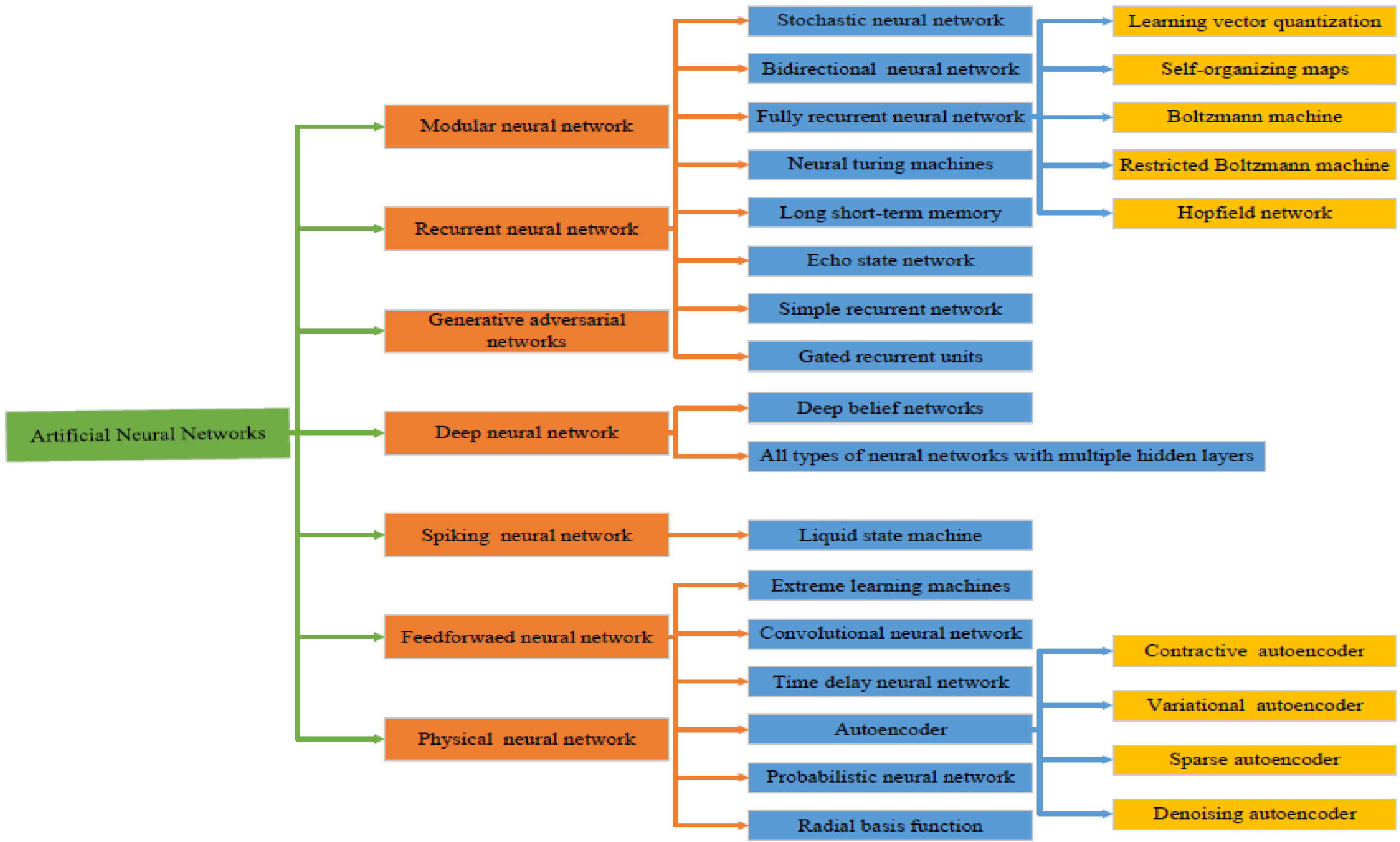
Why now?

•***Machine-learning algorithms*** have progressed in recent years, especially through the development of deep learning and reinforcement-learning techniques based on neural networks.

•***Computing capacity*** has become available to train larger and more complex models much faster. Graphics processing units (GPUs), originally designed to render the computer graphics in video games, have been repurposed to execute the data and algorithm crunching required for machine learning at speeds many times faster than traditional processor chips. *Key Trend Emerging*: Specially design chips and Hardware for Machine Learning workloads(Tensor Units).

•**Massive amounts of data** that can be used to train Machine Learning models are being generated, for example through daily creation of billions of images, online click streams, voice and video, mobile locations, and sensors embedded in the Internet of Things devices.

With Big Data, we can learn from OUR past experience and the Past Experience of Others



Where to compute AI?

The exploration versus exploitation dilemma

Our Daily Lives

Individual Intelligence + Collective Intelligence

On-device AI + Cloud AI = Mobile AI

HUAWEI Kirin 970

The World's First Smartphone AI Computing Platform with HiAI Architecture

Leading Process Technology

10nm Process Technology

High Performance NPU

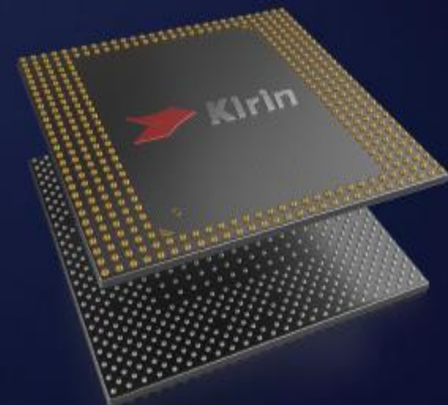
Up to 25x performance

Up to 50x power efficiency

High Performance 8-Core CPU

4xA73 @2.4GHz

4xA53 @1.8GHz



High Efficiency 12-Core GPU

First-to-Market

Mali G72MP12

Advanced Dual ISP

4-Hybrid Focus

Low-light & Motion Shooting

Ultra-Fast 4.5G LTE Modem

4.5G LTE Cat.18 up to

1.2Gbps Download speeds

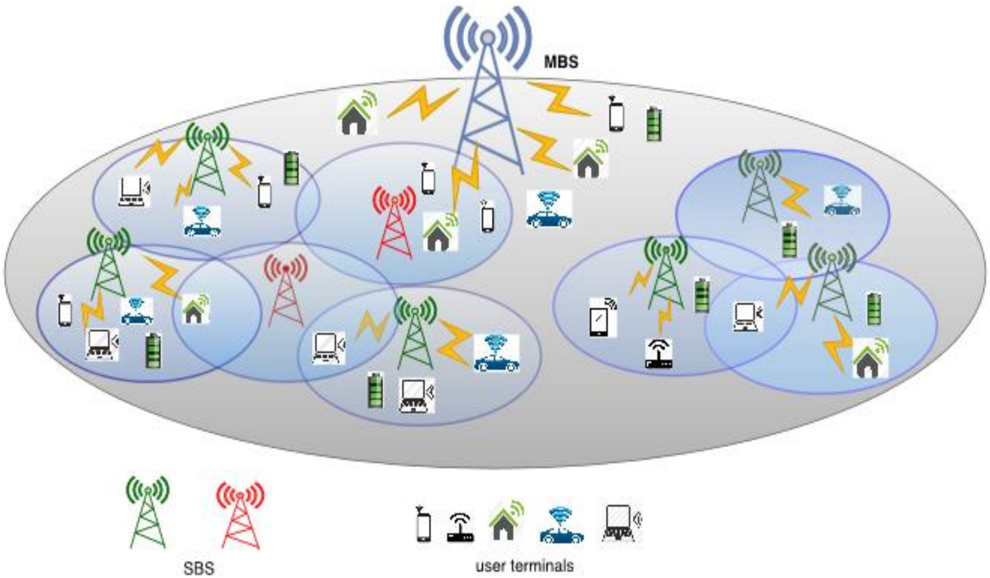
NPU: Neural Network Processing Unit

On Device AI

How to speed up the learning process with local Data? Collaborative AI

- *How, without any detailed modeling of the overall system, can AI based users learn and interact without data exchange?*
- Collaborative AI are based on learning mechanism via imitation that help a user to select its strategy faster by exploiting its local data and the learning outcome of neighboring users.
- In this mechanism, instead of exchanging all the local data between users, only the outcome of their learning algorithms is exchanged.

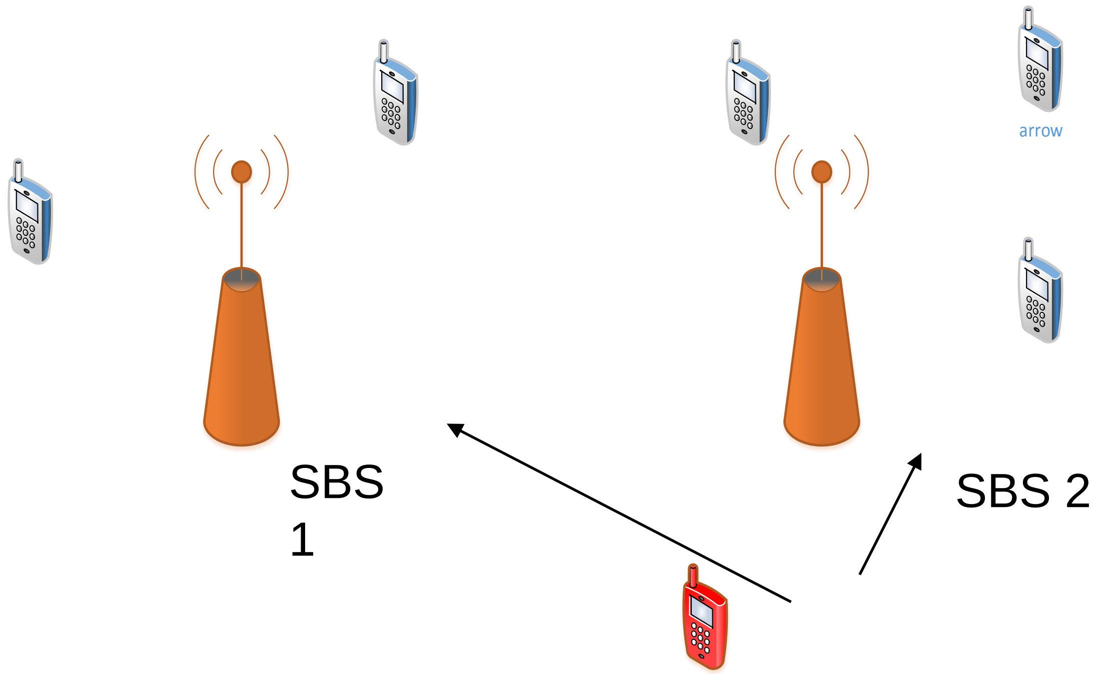
Similarity Learning for User Cell Association



Neighboring devices are in the same conditions.

Similarity Learning for User Cell Association

How to exploit similarity between users to reduce the communication load between users and make the decision process faster?



Similarity learning for User Cell Association

- Each user has to choose the best SBS among its options.
- After selection it will receive a reward based on the congestion level of the SBS and the channel gain between itself and the SBS.
- Each user tries to solve an optimal control problem for selecting the SBS over the time.
- Network is ultra-dense so we model it using mean field games.

Users' Reward Model

Effect of congestion on SBS: a function of the total number of users connected to SBS s

- Users' reward function

$$r_{u,s} = B \log_2 \left(1 + \frac{p_{us} h_{us}}{\sigma^2 + I(\sum_{u'} a_{su'})}\right) - c(\sum_{u'=1}^{U} a_{su'}), \qquad (1)$$

Reward for user u when connected to SBS s

Interference: a function of the total number of users connected to SBS s

Ergodic Control Problem

- Optimization problem for user u located at x:

SBS selected by user u' located at y at time t

Distance between user u located at x and user u' located at y_u

$$\sup_{s(.,\boldsymbol{x})} \lim_{T\to\infty} \inf \frac{1}{T}\sum_{t=0}^{T}\frac{-1}{\epsilon^2}\sum_{u'\in\mathcal{C}_u}\left(s(t,\boldsymbol{x})-s(t,\boldsymbol{y}_i)\right)^2\frac{1}{\epsilon}g\left(\frac{\|\boldsymbol{x}-\boldsymbol{y}_{u'}\|}{\epsilon}\right)\Big)+r_{u,s(t,x)}\Big) \quad (1)$$

Gaussian kernel

Mexican Wave as a Mean Field Equilibrium

$$\inf_{z(t,x)} \liminf_{T\to+\infty} \int_0^T \left(\left[\frac{1}{\epsilon^2} \int_{\mathbb{R}} (z(t,x) - z(t,x-y))^2 \frac{1}{\epsilon} g\left(\frac{y}{\epsilon}\right) dy \right] + F(z(t,x)) + \frac{\dot{z}(t,x)^2}{2} \right) dt$$

where for any $t > 0$ and for any $x \in [0, L]$, $z(t,x) \in [0,1]$ denotes the position at time t of any individual which is sitting into the stadium at the position x ($z = 0$ means that the individual is sitting, $z = 1$ that is standing). Moreover, g is a Gaussian kernel, and the term in square brackets models, roughly speaking, the taste of mimicry of each individual. Finally, F is a given function (is the cost payed if one decides to stay neither sitting nor standing) and the quadratic term is, intuitively, the cost due to the effort of standing up or sitting.

Mean-Field Game (MFG) Formulation

- Hamilton-Jacobi-Bellman equation:

$$V_s^t = \max_{P_s^t \in \mathbb{S}_s(\mathcal{G})} \left\{ r_s(\boldsymbol{\pi}^t, P_s^t) + \sum_{j \in \mathcal{V}_s} P_{sj}^t V_j^{t+1} \right\}$$

vector that represents the transition probability from SBS s to all other SBSs in $\mathbb{S}$

- Fokker-Planck equation (the evolution of users distribution over time):

$$\pi_s^{t+1} = \sum_{j \in \mathcal{V}_s} P_{js}^t \pi_j^t$$

probability that a user connecte to SBS j switches to SBS s at time t

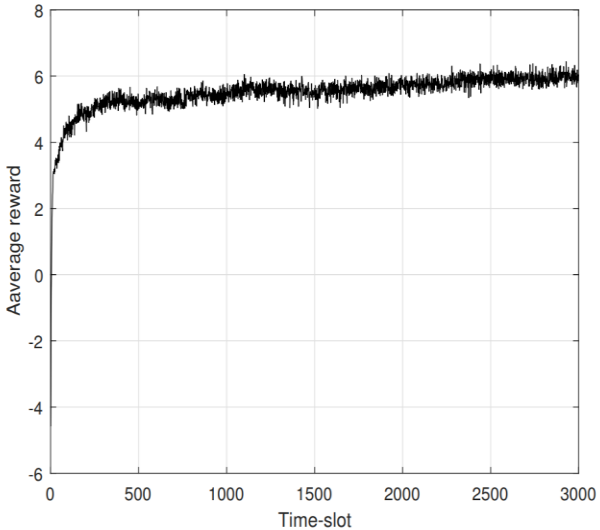
Transforming MFG to MDP

- Solving the mean-field game depends on the form of the utility function which restricts the application domain of MFG
- Given that the MFG problem is difficult to solve, we transform MFG problem into a Markov decision process (MDP).
- For solving MDP, we use deep reinforcement learning.
- Users approximate the value function using adaptive linear neuron (ADALINE) neural networks. The user then trains its network using Widrow-Hoff algorithm (exploration) with the probability ϵ and chooses the best BS with probability $1-\epsilon$.

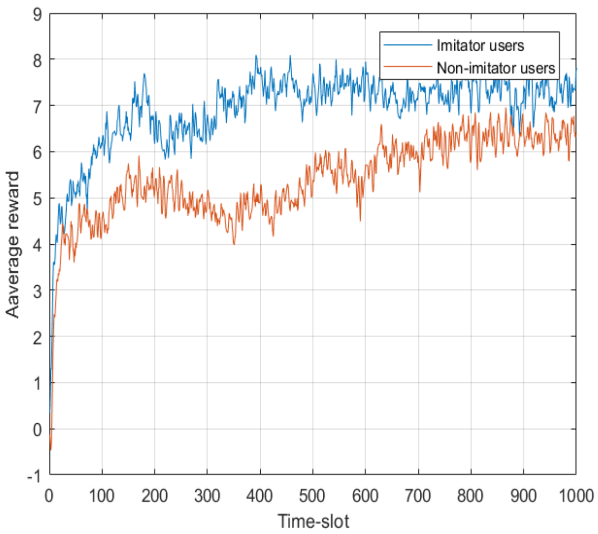
Simulation Results

- We have a system with 10 SBSs in equilibrium
- 20 imitator users and 20 non-imitator users enter the system and start to learn the environment.
- Imitator users can learn the environment faster than non-imitator users.
- Imitator users use the experience of the existing users in the system.

Simulation Results



Equilibrium for the users for the imitator users.

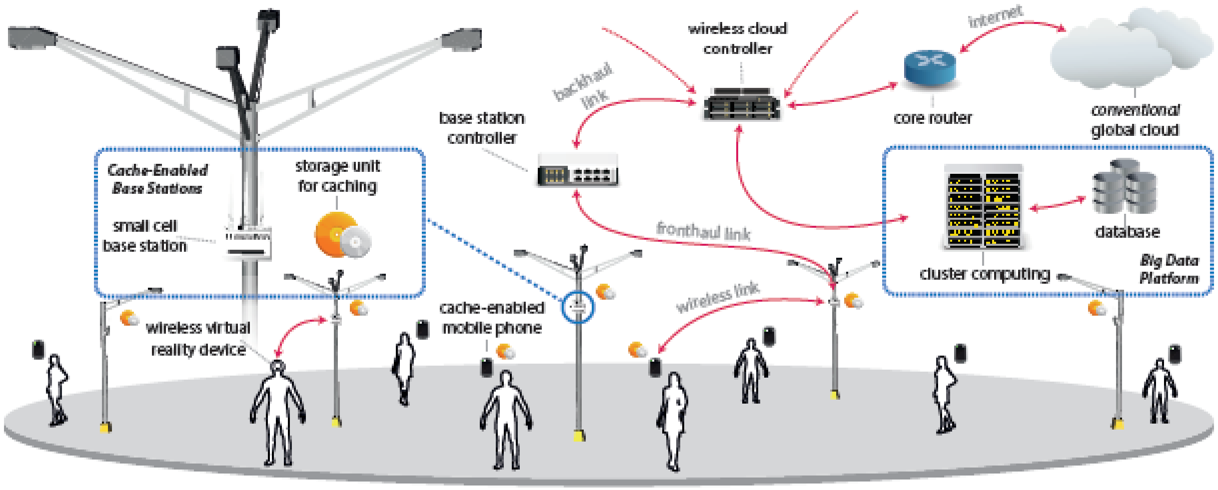


Average reward for 20 imitator users and 20 non-imitator users in the second part.

Cloud AI

Wireless Caching

Overview



An example of wireless cloud architecture.

Page 32

Caching and Machine Learning

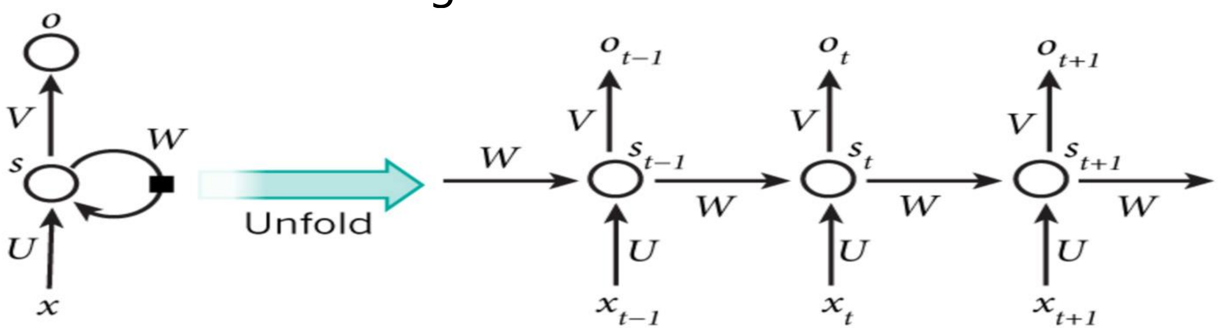
- Challenges in caching systems
  - Base stations (BSs) and devices must have knowledge of each users’ content request and mobility patterns in advance.
  - Users’ content requests determine the contents to cache and users’ mobility patterns determine the users’ association.

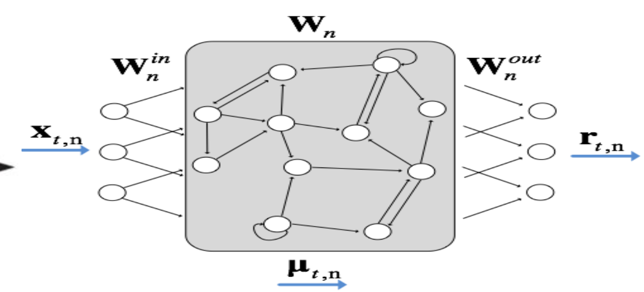
- How to solve?
  - **Machine Learning for the prediction of content request distribution and mobility pattern of each user**

Echo State Networks (ESN)

- ESN is a concept that describes an engineering approach for training and using recurrent neural network.
- Introduced by H. Jaeger [Jae01]
- Why use the ESNs ?
  - Easy to train (only need to train $\mathbf{W}_n^{out}$)
  - Good at dealing with time related data

Herbert Jaeger

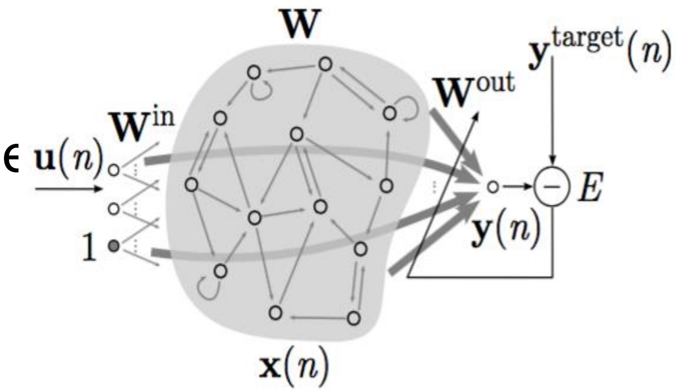




[Jae01] H. Jaeger, “The ‘echo state’ approach to analysing and training recurrent neural networks”, technical report, 2001.

Echo State Networks (ESN)

- Advantage of ESNs
  - Only one hidden layer $\mathbf{W}$
  - Input layer $\mathbf{W}^{\text{in}}$ and hidden layer $\mathbf{W}$ are generated randomly. Only need to train the output layer $\mathbf{W}^{\text{out}}$.
  - Can mimic a target system with arbitrary accuracy.
- Disadvantage of ESNs
  - The Performance of ESN depends on the input layer and hidden layer which are generated randomly. Therefore, it needs to adjust the input and hidden layer to improve ESN.



ESN for content distribution prediction

- Four components:
  - **Agents**: BSs/devices (Each ESN predicts one user's content request distribution)
  - **Inputs**: User's context including content request time, gender, occupation, age, and device type:

$$\boldsymbol{x}_{t,j} = \left[x_{tj1}, \cdots, x_{tjK}\right]^{\mathrm{T}}$$

  - **Outputs**: Content request distribution

$$\boldsymbol{y}_{t,j} = \left[p_{tj1}, p_{tj2}, \ldots, p_{tjN}\right]$$

  - **ESN Model**: Input weight matrix $\mathbf{W}_j^{a,in}$

    Recurrent matrix $\mathbf{W}_j^{a}$

    Output weight matrix $\mathbf{W}_j^{a,out}$

ESN for content distribution prediction

- Training:
  - Dynamic reservoir state update (record the ESN history data):

$$\boldsymbol{v}_{t,j}^{\alpha} = f\left(\boldsymbol{W}_{j}^{\alpha}\boldsymbol{v}_{t-1,j}^{\alpha} + \boldsymbol{W}_{j}^{\alpha,in}\boldsymbol{x}_{t,j}\right),$$

  - Output:

$$\boldsymbol{y}_{t,j}\left(\boldsymbol{x}_{t,j}\right) = \boldsymbol{W}_{t,j}^{\alpha,out}\left[\boldsymbol{v}_{t,j}^{\alpha};\boldsymbol{x}_{t,j}\right],$$

  - Output weight matrix update

$$\boldsymbol{W}_{t+1,j}^{\alpha,out} = \boldsymbol{W}_{t,j}^{\alpha,out} + \lambda^{\alpha}\left(\boldsymbol{e}_{t,j} - \boldsymbol{y}_{t,j}\left(\boldsymbol{x}_{t,j}\right)\right)\left[\boldsymbol{v}_{t,j}^{\alpha};\boldsymbol{x}_{t,j}\right]^{\mathrm{T}},$$

ESN for mobility pattern prediction

- Four components:
  - **Agents**: BSs (Each ESN predicts one user's content request distribution)
  - **Inputs**: Current location $m_{t,j}$ (ESN can find the relationship with historical locations)
  - **Output**: Predicted location:

$$\boldsymbol{s}_{t,j} = \left[s_{tj1}, \cdots, s_{tjN_s}\right.$$

$$\boldsymbol{W}_j = \begin{bmatrix} 0 & 0 & \cdots & w \\ w & 0 & 0 & 0 \\ 0 & \ddots & 0 & 0 \\ 0 & 0 & w & 0 \end{bmatrix},$$

  - **ESN Model**: Recurrent matrix

    Input weight matrix $\mathbf{W}_j^{in}$

    Output weight matrix $\mathbf{W}_j^{out}$

ESN for mobility pattern prediction

- Training:
  - Dynamic reservoir state update (record the ESN history data):

$$\boldsymbol{v}_{t,j} = \boldsymbol{W}_j \boldsymbol{v}_{t-1,j} + \boldsymbol{W}_j^{in} \boldsymbol{m}_{t,j}$$

  - Output:

$$\boldsymbol{s}_{t,j} = \boldsymbol{W}_j^{out} \boldsymbol{v}_{t,j}$$

  - Output weight matrix update

$$\boldsymbol{W}_j^{out} = \boldsymbol{s}_j \boldsymbol{v}_j^{\mathrm{T}} \left(\boldsymbol{v}_j^{\mathrm{T}} \boldsymbol{v}_j + \lambda^2 \mathbf{I} \right)^{-1}$$

Algorithm

Algorithm 1 Algorithm with ESNs and sublinear algorithms

Input: The set of users' contexts, $\boldsymbol{x}_t$ and $\boldsymbol{m}_t$;

Init: initialize $\boldsymbol{W}_j^{\alpha,in}$, $\boldsymbol{W}_j^{\alpha}$, $\boldsymbol{W}_j^{\alpha,out}$, $\boldsymbol{W}_j^{in}$, $\boldsymbol{W}_j$, $\boldsymbol{W}_j^{out}$, $\boldsymbol{y}_j = 0$, $\boldsymbol{s}_j = 0$, ϵ, and δ

for time T_τ **do**
    update the output weight matrix $\boldsymbol{W}_{T_\tau+1,j}^{out}$ based on (17)
    obtain prediction $\boldsymbol{s}_{T_\tau+1,j}$ based (16)
    for time τ **do**
        obtain prediction $\boldsymbol{y}_{\tau+1,j}$ based on (12)
        update the output weight matrix $\boldsymbol{W}_{\tau+1,j}^{\alpha,out}$ based on (13)
        determine which content to cache in each RRH based on (20)
        cluster the RRHs
    end for
    calculate the content percentage for each content based on (22)
    determine which content to cache in cloud based on (21)
end for

Simulation settings

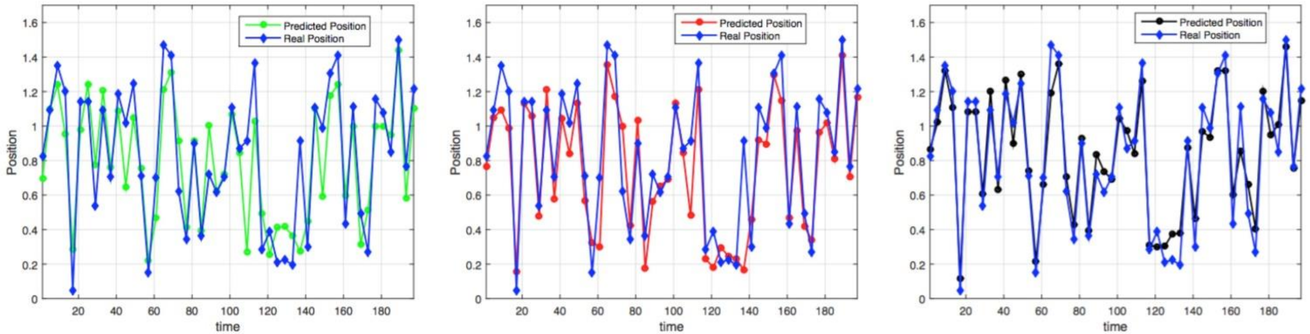
TABLE I
SYSTEM PARAMETERS

Parameters	Values	Parameters	Values
r	1000 m	P	20 dBm
R	1000	β	4
B	1 MHz	λ^{α}	0.01
L	10 Mbit	S	25
θ_s^O	0.05	T	300
N_w	1000	σ^2	-95 dBm
C_c,C_r	6,3	$D_{\max}$	1
K	7	N_s	10
δ	0.05	ϵ	0.05
H	3	λ	0.5
T_τ	30	χ	0.85

- *Real data from Youku* for content simulations and use the *realistic measured mobility data from the Beijing University of Posts and Telecommunications* for mobility simulations.

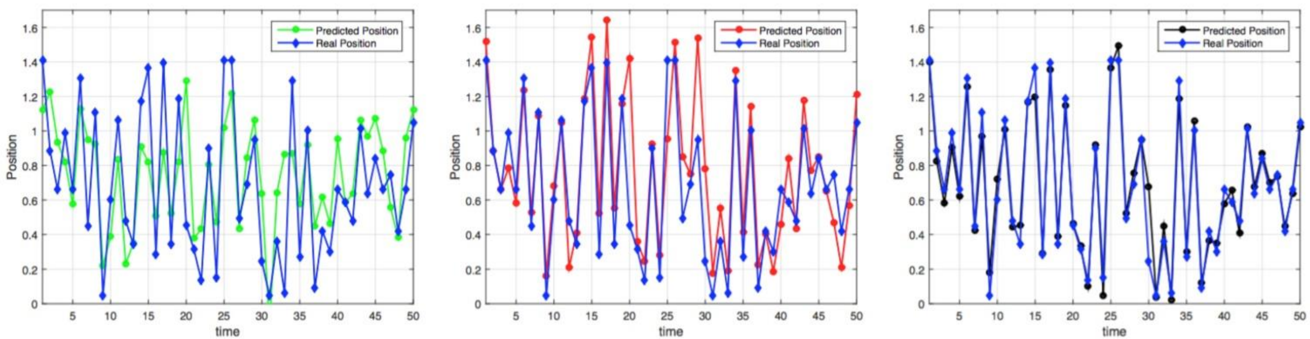
- M. Chen, W. Saad, C. Yin, and M. Debbah, "Echo State Networks for Proactive Caching in Cloud-Based Radio Access Networks with Mobile Users", accepted for publication, IEEE Transactions on Wireless Communications, 2017

Simulation results



(a) $N_{tr} = 4500$ (b) $N_{tr} = 7500$ (c) $N_{tr} = 10500$

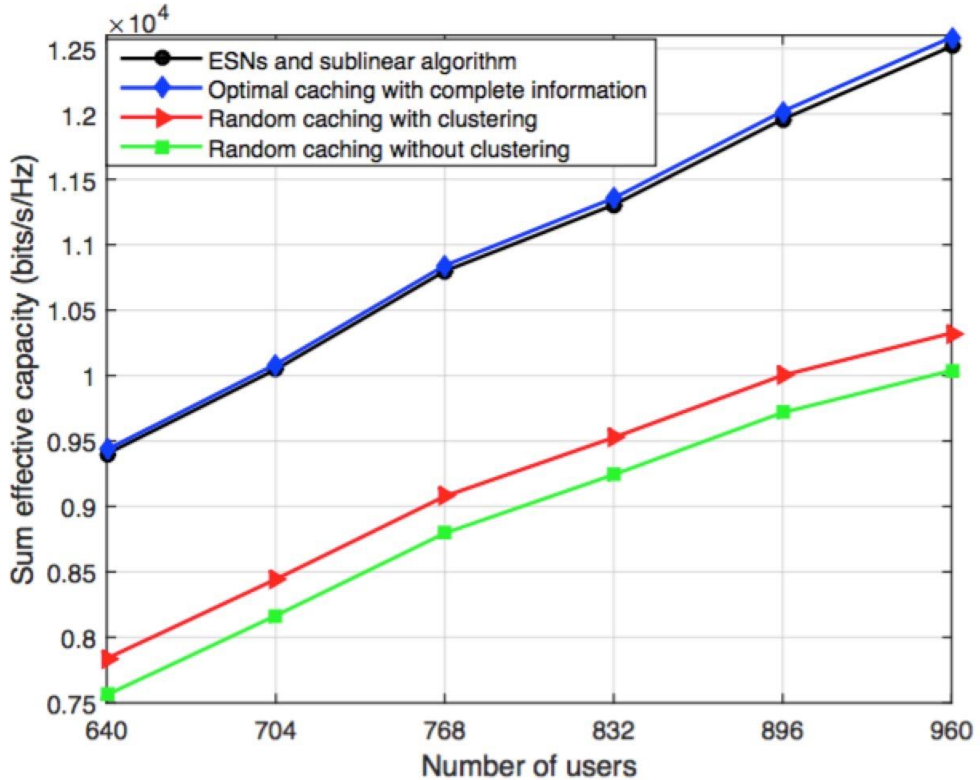
The ESNs prediction of the users mobility as the training dataset N_{tr} varies.



(a) $W = 300$ (b) $W = 800$ (c) $W = 3000$

The ESNs prediction of the users mobility as the ESNs reservoir units varies.

Simulation results



Sum effective capacity vs. the number of the users.

Thank You.