ITU-ML5G-PS-004: Federated Learning for Spatial Reuse in a multi-BSS (Basic Service Set) scenario

Universitat Pompeu Fabra Barcelona

Francesc Wilhelmi

8 September 2021

Table of contents

Outline

1. Introduction
2. Background
3. Federated Learning for Spatial Reuse
4. Introduction to the Dataset

Important information

Important resources:

- Challenge website: https://aiforgood.itu.int/about/aiml-in-5g-challenge/
- PS website: https://www.upf.edu/web/wnrg/2021-edition
- Dataset: https://zenodo.org/record/5352060#.YTd3mtMzba1

Registration (at the official website):

1. Choose the problem statement ITU-ML5G-PS-004
2. The team leader creates a team (needs approval from admin)
3. Team members request to join the team (needs approval from team leader)

Timeline

February – May 2021

Curation Phase

June – October 2021

Competition Phase

- June – September 2021 – Registration
- Hard registration deadline 14 September 2021
- June – Sept/beginning of October 2021 – Submission of solutions
- October 2021 – Evaluation of solutions
- October – November 2021 – Preparation of reports

December 2021

Grand Challenge Finale (online)

- November 2021 – Judges Panel evaluates the best solutions from Competition Phase
- Best solutions pitch in a 2-day event end of November 2021 to determine the finalists
- December 2021 – Grand Challenge Finale

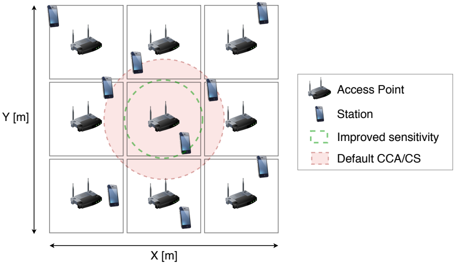
Registration deadline: 14 September!

Outline

Spatial Reuse

Basics

- **Goal:** ignore inter-BSS transmissions
- **Means:** tuning the CCA threshold
- **Constraints:** transmit power limitation



- Video-presentation of the IEEE 802.11ax SR topic
- Tutorial paper on the topic: https://arxiv.org/abs/1907.04141

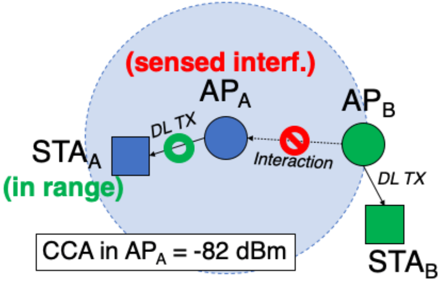
The CCA threshold

CSMA/CA Operation

- Implement decreasing random backoff before transmitting
- Perform physical carrier sensing to assess whether the medium is busy or not
- Apply CCA mechanism:
  1. Check signal source (Wi-Fi or non-Wi-Fi)
  2. Apply threshold (e.g., -82 dBm)

For simplicity, we use the CCA as the unique threshold for detecting the channel busy/idle.

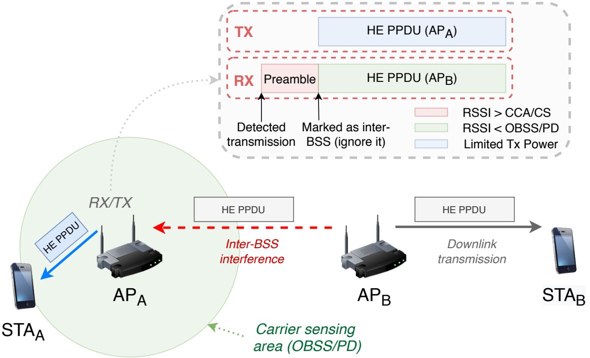
Effects of tuning the CCA threshold



Tx Power fixed to 20 dBm

Spatial Reuse in IEEE 802.11ax

- Two mechanisms:
  1. *OBSS/PD-based SR*
  2. *Parametrized SR*
- Common features: fast source identification, sensitivity adjustment, tx power limitation



Federated Learning (I)

Google AI Blog

The latest news from Google AI

Federated Learning: Collaborative Machine Learning without Centralized Training Data

Thursday, April 6, 2017

Posted by Brendan McMahan and Daniel Ramage, Research Scientists

Standard machine learning approaches require centralizing the training data on one machine or in a datacenter. And Google has built one of the most secure and robust cloud infrastructures for processing this data to make our services better. Now for models trained from user interaction with mobile devices, we're introducing an additional approach: *Federated Learning*.

Federated Learning

- Introduced by Google*
- Decentralized data distribution
- Some features:
  - High scalability
  - Fault-tolerant
  - Privacy
  - Suitable for non-IID data
  - Specialized training

*https://ai.googleblog.com/2017/04/federated-learning-collaborative.html

Federated Learning (II)

$$\min_{w\in\mathbb{R}^d} \sum_{k=1}^{K} \frac{n_k}{n} F_k(w),$$

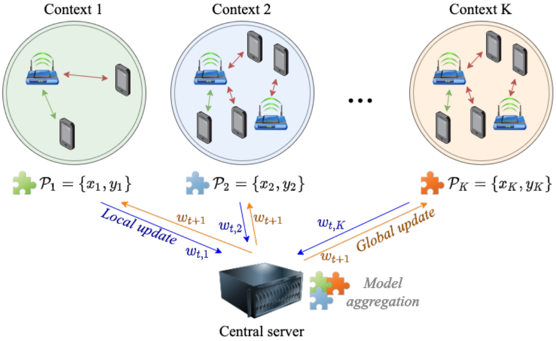
- w: model parameters to be jointly optimized
- n_k: data points on client k
- n: total data points
- F_k: loss function on client k ($F_k = \frac{1}{n_k}\sum_{i=1}^{n_k} f_i(w)$)

Outline

The problem statement in a nutshell

Federated Learning for Spatial Reuse

- **Data**: simulated IEEE 802.11ax SR scenarios
- **Goal:** develop FL algorithms to predict performance
- **Evaluation:** test dataset (new scenarios)



Recommended steps

1. Process the dataset and extract the features of interest (e.g., interference, RSSI, SINR)
2. Split data from different contexts (i.e., simulation scenarios) $\rightarrow$ make data "federated"
    - Each context has been simulated for different configurations
    - All the tried combinations correspond to a single FL "client"
3. Train the FL algorithm by aggregating the model updates provided by each context, which perform local training functions

Federated Learning resources

- TensorFlow FL libraries are recommended, but other FL solutions are accepted (e.g., Pytorch, custom solutions).
  - Install TensorFlow Federated (TFF) → here
  - Video tutorial on TFF → here.
- Choose the training model (e.g., SGD)
- Choose the aggregation approach (e.g., FedAvg)

```
import tensorflow as tf
import tensorflow_federated as tff

# Load simulation data.
source, _ = tff.simulation.datasets.emnist.load_data()
def client_data(n):
  return source.create_tf_dataset_for_client(source.client_ids[n]).map(
      lambda e: (tf.reshape(e['pixels'], [-1]), e['label'])
  ).repeat(10).batch(20)

# Pick a subset of client devices to participate in training.
train_data = [client_data(n) for n in range(3)]

# Grab a single batch of data so that TFF knows what data looks like.
sample_batch = tf.nest.map_structure(
    lambda x: x.numpy(), iter(train_data[0]).next())

# Wrap a Keras model for use with TFF.
def model_fn():
  model = tf.keras.models.Sequential([
      tf.keras.layers.Dense(10, tf.nn.softmax, input_shape=(784,),
                            kernel_initializer='zeros')
  ])
  return tff.learning.from_keras_model(
      model,
      dummy_batch=sample_batch,
      loss=tf.keras.losses.SparseCategoricalCrossentropy(),
      metrics=[tf.keras.metrics.SparseCategoricalAccuracy()])

# Simulate a few rounds of training with the selected client devices.
trainer = tff.learning.build_federated_averaging_process(
  model_fn,
  client_optimizer_fn=lambda: tf.keras.optimizers.SGD(0.1))
state = trainer.initialize()
for _ in range(5):
  state, metrics = trainer.next(state, train_data)
  print (metrics.loss)
```

Outline

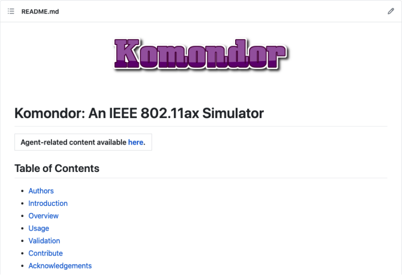
Generating synthetic training datasets

The Komondor simulator

- IEEE 802.11ax-oriented discrete-event simulator
- Fast performance & ML

Usage

- Simulate OBSS/PD-based SR
- Large-scale deployments
- Complete datasets hard to get from measurements


README.md

Komondor

Komondor: An IEEE 802.11ax Simulator

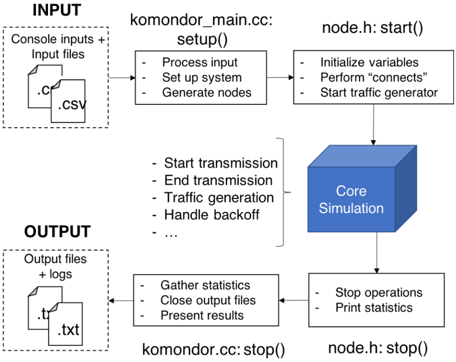
Agent-related content available here.

Table of Contents

- Authors
- Introduction
- Overview
- Usage
- Validation
- Contribute
- Acknowledgements

Open-source project: https://github.com/wn-upf/Komondor

Dataset generation



Simulated sce. (for now)

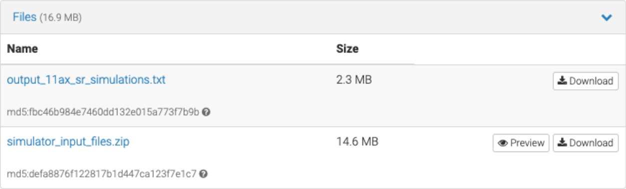
- 1,000 random contexts (2-6 APs, 1 STA per AP)
- 21 possible conf. (-82 to -62 dBm) per context
- Distance constraints (e.g., 10 m between APs)
- UDP downlink traffic
- 10 sec. simulations

Commit: 172eceb

Dataset overview

Files

- Main file: “output_11ax_sr_simulations.txt”
- Complementary files: “simulator_input_files.zip”


Files (16.9 MB)

Name	Size	
output_11ax_sr_simulations.txt md5:fbc46b984e7460dd132e015a773f7b9b	2.3 MB	Download
simulator_input_files.zip md5:defa8876f122817b1d447ca123f7e1c7	14.6 MB	Preview Download

Features and labels in the main file

Files

1. Header line
2. Per-STA throughput
3. Inter-AP interference
4. Per-STA RSSI
5. Per-STA average SINR

output_11ax_sr_simulations.txt

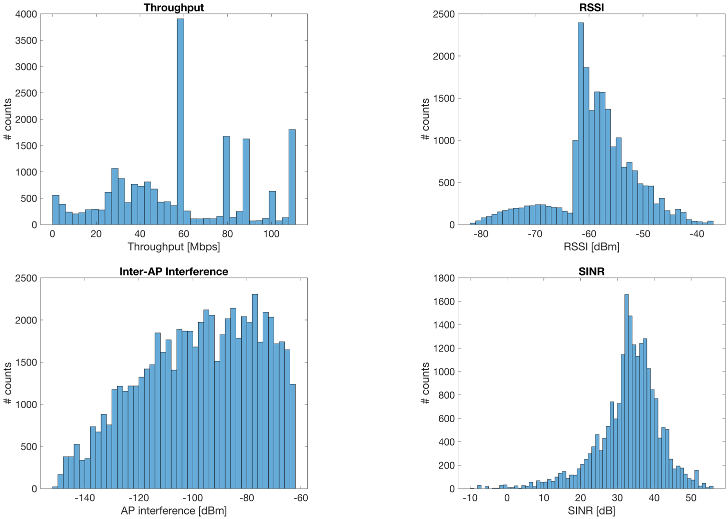
```
 KOMONDOR SIMULATION 'sim_input_nodes_s0993_c-82.csv' (seed 1991)
47.63
-118.31,-128.99,-101.62,-113.05,-64.19
-56.95
36.45
 KOMONDOR SIMULATION 'sim_input_nodes_s0994_c-62.csv' (seed 1991)
33.60
-90.82,-67.10,-103.81,-85.41,-84.88
-60.39
24.35
 KOMONDOR SIMULATION 'sim_input_nodes_s0994_c-63.csv' (seed 1991)
33.60
-90.82,-67.10,-103.81,-85.41,-84.88
-60.39
24.35
 KOMONDOR SIMULATION 'sim_input_nodes_s0994_c-64.csv' (seed 1991)
33.63
-90.82,-67.10,-103.81,-85.41,-84.88
-77.39
24.39
 KOMONDOR SIMULATION 'sim_input_nodes_s0994_c-65.csv' (seed 1991)
33.30
-90.82,-67.10,-103.81,-85.41,-84.88
```

Input files

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	node_code	node_type	wlan_code	x(m)	y(m)	z(m)	central_freq(	channel_bon	primary_cha	min_channel	max_channe	tpc_default(	cca_default(
2	AP_A	0	A	10	10	0	5.0	4	4	4	5	20	-82
3	STA_A1	1	A	0.0713	108.079	0	5.0	4	4	4	5	20	-82
4	STA_A2	1	A	19.627	41.427	0	5.0	4	4	4	5	20	-82
5	STA_A3	1	A	137.849	167.538	0	5.0	4	4	4	5	20	-82
6	STA_A4	1	A	67.112	17.487	0	5.0	4	4	4	5	20	-82
7	STA_A5	1	A	131.934	23.628	0	5.0	4	4	4	5	20	-82
8	STA_A6	1	A	176.857	76.662	0	5.0	4	4	4	5	20	-82
9	STA_A7	1	A	194.473	84.359	0	5.0	4	4	4	5	20	-82
10	STA_A8	1	A	43.802	20.739	0	5.0	4	4	4	5	20	-82

Simulator input files are complementary to the main data file

Some insights



Final remarks

- Participants must use the provided dataset to train an FL algorithm
- The output of the ML algorithm should be a throughput prediction
- The choice of the ML approach is decided by each participant (neural network, linear regression, decision tree, etc.)
- A test dataset will be provided to evaluate the performance of the proposed models
- Three deliverables:
  1. FL model and predictions on the test dataset (25 October 2021)
  2. Documentation + Report (7 November 2021)
  3. Elevator pitch (November 2021)
  4. Final presentation in the grand finale (December 2021)
- The winners will be invited to collaborate in a publication

Questions

?

Francesc Wilhelmi, Ph.D.
fwilhelmi@cttc.cat
Centre Tecnològic de Telecomunicacions de Catalunya (CTTC)