IT-UNIVERSITETET I KØBENHAVN

EMISSIONS TRACKING: COMPUTE THE EMISSIONS OF YOUR PROJECTS / YOUR TRAINING SESSIONS

Ambroise LAROYE--LANGOUËT

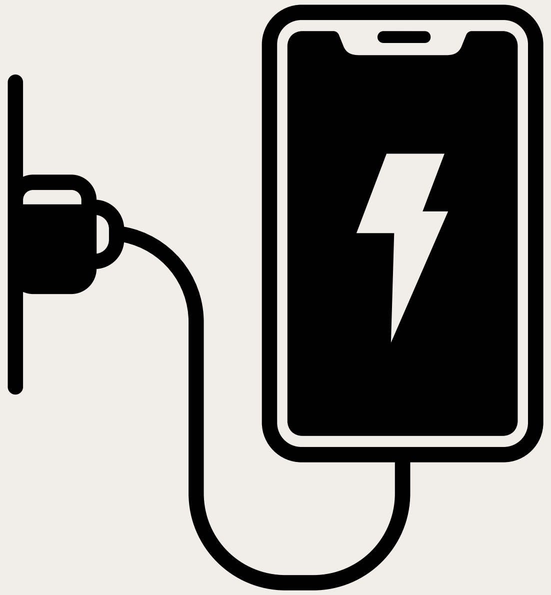
CODE CARBON

ENERGY COST

INTRODUCTION

According to the IEA, a single Google search takes 0.3 watt-hours of electricity, while a ChatGPT request takes 2.9 watt-hours.

Thus a single ChatGPT prompt is estimated to consume, on average, as much energy as 40 mobile phone charges



x 40

POWER USAGE : What are the main consumers of energy in computers?

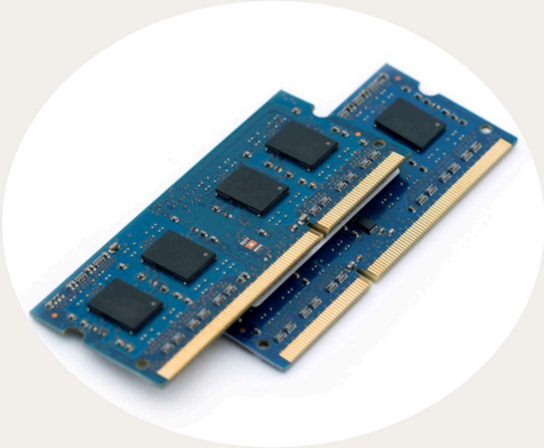
SSD/HDD

HDD and SSD (Solid State Drives) are the permanent storage devices on a PC.
SSDs consume less energy than traditional hard drives (HDDs). Generally, SSDs consume around 2 to 5 watts in operation, while HDDs can consume from 6 to 15 watts.

Lower power consumer

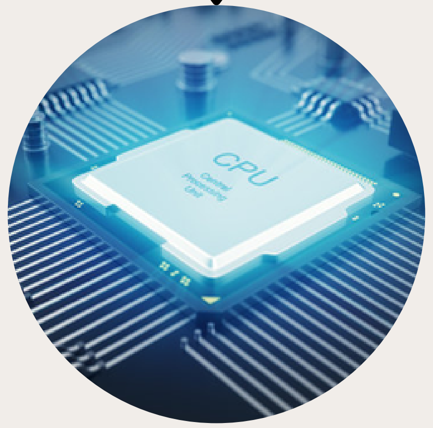
Other components

such as : the motherboard, cooling fans, and power supply units (PSUs).
PSUs can be very efficient or less efficient, which can affect the overall system consumption.



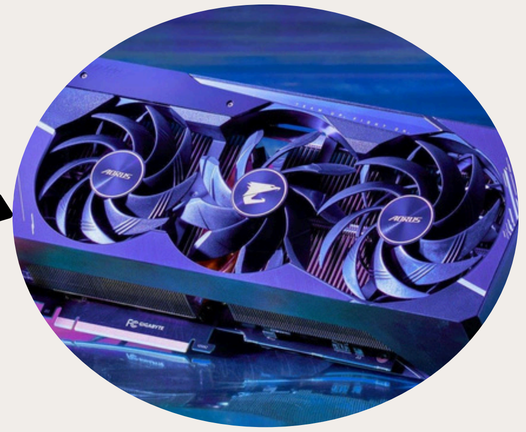
RAM (Random Access Memory)

RAM is used to temporarily store data during training.
The energy consumption of RAM is lower than that of GPUs and CPUs but remains significant, especially in systems with large amounts of RAM (32 GB and more). Typical consumption can vary from 5 to 15 watts per module.



CPU

It is the brain of the computer. It executes program instructions and manages the computer's main tasks. They have several cores, enabling them to handle several tasks simultaneously.
CPUs can consume between 65 and 150 watts depending on the model and workload.



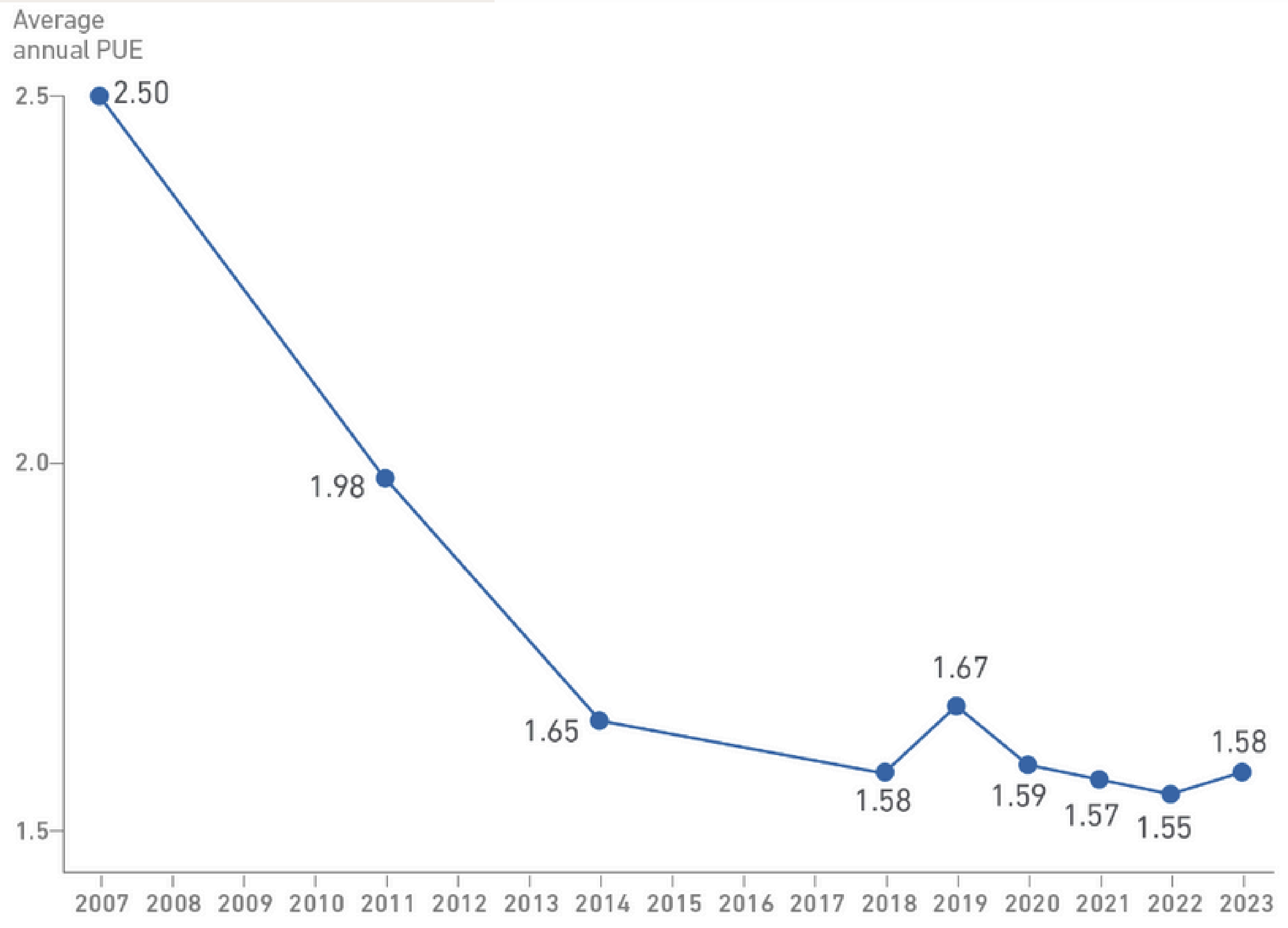
GPU (Graphics Processing Unit)

GPUs are the main drivers of neural network training due to their ability to perform massive parallel computations.
They consume a lot of energy, often between 200 and 400 watts per card, and can exceed this range for high-end models used in data centers

POWER - ENERGY

- Relation between power and energy : $E = \int_0^t p(t)dt$
- The resulting average power (in Watts) is defined as : $P_{avg} = \frac{E}{T}$

The energy E (in Joules) is defined as the effort to perform a task during a certain period of time T (in seconds)



We have access to power and energy metrics in some areas of electronics (e.g., with powermeters), measuring the energy consumption of any kind of computer program is already a challenging task, since there are many variables involved (e.g. cache hits, cache misses, DRAM accesses).
We can approximate the energy and carbon cost by picking up the metrics of : GPU (Pg), CPU (Pc) and RAM (Pr).
To achieve this, we are using : NVIDIA System Management Interface and the Intel's Running Average Power Limit.

$$\boxed{P_{avg} = PUE * (nP_g + P_c + P_r)}$$

with :

- Power Usage Effectiveness (PUE) coefficient estimates additional energies required to sustain the computing infrastructure (mainly cooling)
  PUE = 1.58 (as it is the 2023 global average for data centers)
- n is the total number of GPUs used

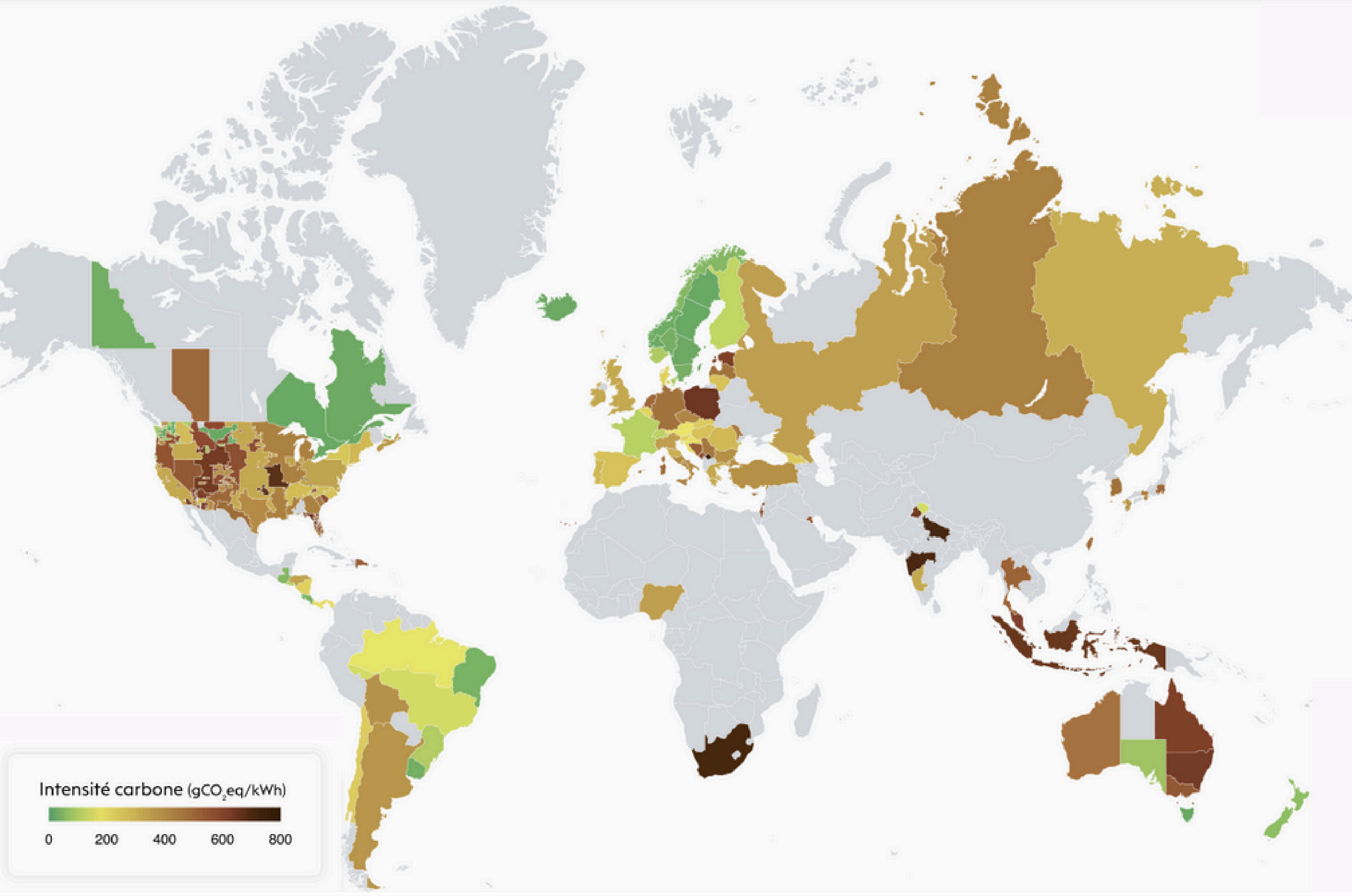
Reminder : **1 kWh = 3600 kJ**

POWER - CARBON COST

$$P_{avg} = PUE * (gP_g + P_c + P_r)$$

The carbon emission intensity factor (in kgCO2eq/kWh).

This factor is location dependent, but can be captured in real-time :
https://app.electricitymaps.com/map



The U.S. Environmental Protection Agency (EPA) provides average CO2 produced (in pounds per kilowatt-hour) for power consumed in the U.S., which we use to convert power to estimated CO2 emissions :

$$CO_2e = 0.954 * P_{avg}$$

FLOPS (Floating point operations per second):

FLOPS are a unit for evaluating the computing power of computers. They are used to compare different processing architectures and assess their ability to handle computationally intensive tasks.
Although there is a correlation between power and FLOPS, FLOPS is not a measure of energy consumption, but of the performance of HPC servers.

$$FLOPS = Cores * Frequency * FLOPS/Cycle$$

with :

- Cores : number of cores used
- Frequency : clock frequency (Graphics clock) in GHz
- FLOPS/Cycle (see further)

HIGH PERFORMANCE COMPUTING (HPC) AT ITU

```
(base) [[redacted]@cn5 [redacted]]$ nvidia-smi -q

==============NVSMI LOG==============

Timestamp                                 : Wed Jul  3 17:47:50 2024
Driver Version                            : 525.89.02
CUDA Version                              : 12.0

Attached GPUs                             : 1
GPU 00000000:82:00.0
    Product Name                          : Tesla V100-PCIE-32GB
    Product Brand                         : Tesla
    Product Architecture                  : Volta
    Display Mode                          : Enabled
    Display Active                        : Disabled
    Persistence Mode                      : Disabled
    MIG Mode
        Current                           : N/A
        Pending                           : N/A
    Accounting Mode                       : Disabled
    Accounting Mode Buffer Size           : 4000
    Driver Model
        Current                           : N/A
        Pending                           : N/A
    Serial Number                         : 1563419019623
    GPU UUID                              : GPU-f37942a0-22ec-d681-0985-5c750262f95b
    Minor Number                          : 1
    VBIOS Version                         : 88.00.7E.00.03
    MultiGPU Board                        : No
    Board ID                              : 0x8200
    Board Part Number                     : 900-2G500-0010-000
    GPU Part Number                       : 1DB6-897-A1
    Module ID                             : 1
```

Number of CUDA Cores = 5120

$$FLOPS = Cores * Frequency * \frac{GPU\,frequency}{NumberCUDACores * InstructionsPerCycles}$$

Frequency = 135 MHz

```
Clocks
    Graphics                              : 135 MHz
    SM                                    : 135 MHz
    Memory                                : 877 MHz
    Video                                 : 555 MHz
```

HPC AT ITU

http://hpc.itu.dk/

Node	CPU	Cores	Memory	GPU	Interconnect	Tmp space
front	CPU	32	256 GiB	None	100 Gbps IB	400 GiB
cn1	CPU	32	256 GiB	None	100 Gbps IB	3600 GiB
cn2	CPU	32	256 GiB	None	100 Gbps IB	3600 GiB
cn3	CPU	48	192 GiB	2x Tesla V100	100 Gbps IB	3600 GiB
cn4	CPU	48	192 GiB	4x Tesla V100	100 Gbps IB	3600 GiB
cn5	CPU	32	512 GiB	2x Tesla V100	100 Gbps IB	3400 GiB
cn6	CPU	64	384 GiB	6x Tesla V100	100 Gbps IB	3600 GiB
cn8	CPU	256	256 GiB	None	100 Gbps IB	400 GiB
desktop1	CPU	8	32 GiB	1x RTX 2070	1 Gbps Eth	400 GiB
desktop2	CPU	8	32 GiB	1x RTX 2070	1 Gbps Eth	400 GiB
desktop3	CPU	8	32 GiB	1x RTX 2070	1 Gbps Eth	400 GiB
desktop4	CPU	8	32 GiB	1x RTX 2070	1 Gbps Eth	400 GiB
desktop5	CPU	8	32 GiB	1x RTX 2070	1 Gbps Eth	400 GiB
desktop6	CPU	8	32 GiB	1x RTX 2070	1 Gbps Eth	400 GiB
desktop7	CPU	8	32 GiB	1x RTX 2070	1 Gbps Eth	400 GiB
desktop8	CPU	8	32 GiB	1x RTX 2070	1 Gbps Eth	400 GiB
desktop9	CPU	8	32 GiB	1x RTX 2070	1 Gbps Eth	400 GiB
desktop10	CPU	8	32 GiB	1x RTX 2070	1 Gbps Eth	400 GiB
desktop11	CPU	8	32 GiB	None	1 Gbps Eth	400 GiB
desktop12	CPU	8	32 GiB	None	1 Gbps Eth	400 GiB
desktop13	CPU	8	32 GiB	None	1 Gbps Eth	400 GiB
desktop14	CPU	8	32 GiB	None	1 Gbps Eth	400 GiB
desktop15	CPU	8	32 GiB	None	1 Gbps Eth	400 GiB
desktop16	CPU	8	32 GiB	None	1 Gbps Eth	400 GiB
desktop17	CPU	80	394 GiB	2x RTX 8000	1 Gbps Eth	400 GiB
desktop18	CPU	96	256 GiB	4x Nvidia A30 1x Nvidia A100 80GiB	1 Gbps Eth	400 GiB
desktop19	CPU	48	120 GiB	None	1 Gbps Eth	400 GiB
desktop20*	CPU	48	120 GiB	None	1 Gbps Eth	400 GiB
desktop21	CPU	40	256 GiB	2x GTX 1080ti	1 Gbps Eth	400 GiB
desktop22	CPU	96	120 GiB	4x Nvidia A100 40GiB	1 Gbps Eth	400 GiB
desktop23	CPU	40	250 GiB	2x GTX 1080ti	1 Gbps Eth	400 GiB
desktop24	CPU	40	250 GiB	2x GTX 1080ti	1 Gbps Eth	400 GiB
desktop25	CPU	96	250 GiB	2x Nvidia A30	1 Gbps Eth	400 GiB
desktop26	CPU	192	250 GiB	2x Nvidia A30	1 Gbps Eth	400 GiB
desktop27	CPU	48	250 GiB	2x Nvidia RTX 6000	1 Gbps Eth	400 GiB

TUTORIAL : CODECARBON

01 Add this to your code

```
from codecarbon import EmissionsTracker


tracker = EmissionsTracker() # or EmissionsTracker(project_name="...")
tracker.start()
#
# Compute intensive code goes here
#
tracker.stop()
```

More details in my repository :
https://github.com/Ambroise012/EnergyTracker

02 Get your results in the emissions.csv file

duration	emissions	emissions_rate	cpu_power	gpu_power	ram_power	cpu_energy	gpu_energy	ram_energy
106.18328142166138	2.9586103545726807e-05	2.786324094490759e-07	125.2348312083332	232.68873551630367	70.27894020080566	0.00372537428557748	0.0066481503185159	0.0020721558207669
106.30764317512512	2.977536686002682e-05	2.800867931102243e-07	125.75981502958363	232.42140223798503	70.27894020080566	0.0037714373015918	0.0066792383989419	0.0020746201668334
55.26943397521973	1.4813553105451898e-05	2.680243317145896e-07	128.49642534932198	210.3570493915212	70.27894020080566	0.0019755201029148	0.003177307264066	0.0010786369291597
44.43157124519348	1.1654073084097516e-05	2.6229261665731055e-07	128.46890076822473	202.2093859065853	70.27894020080566	0.0015804946885614	0.002454840297204	0.0008670635010256
32.01623201370239	8.12943067425307e-06	2.539159096165288e-07	127.14778303656466	192.2612447501428	70.27894020080566	0.0011355796492963	0.001659526605398	0.0006246173345041
21.63544464111328	5.102500109674281e-06	2.358398541982415e-07	125.81523303134588	163.95354411120195	70.27894020080566	0.0007707252501908	0.0009537413185479	0.0004219494212768
18.05564785003662	3.987396470843711e-06	2.2083929106069841e-07	129.12095539193473	140.4555492020566	70.27894020080566	0.0006441511219871	0.0006809019336099	0.0003522838105645

03 Display the total energy used during your project, from the csv file

```
carbonboard --filepath="emissions.csv" --port=3333
```

Infrastructure Hosted at nan, Denmark

Power Consumption Across All Experiments : 47.3 kWh

Carbon Equivalent Across All Experiments : **7.2 kg**

Last Run Power Consumption : 0.0 kWh

Last Run Carbon Equivalent : 0.0 kg

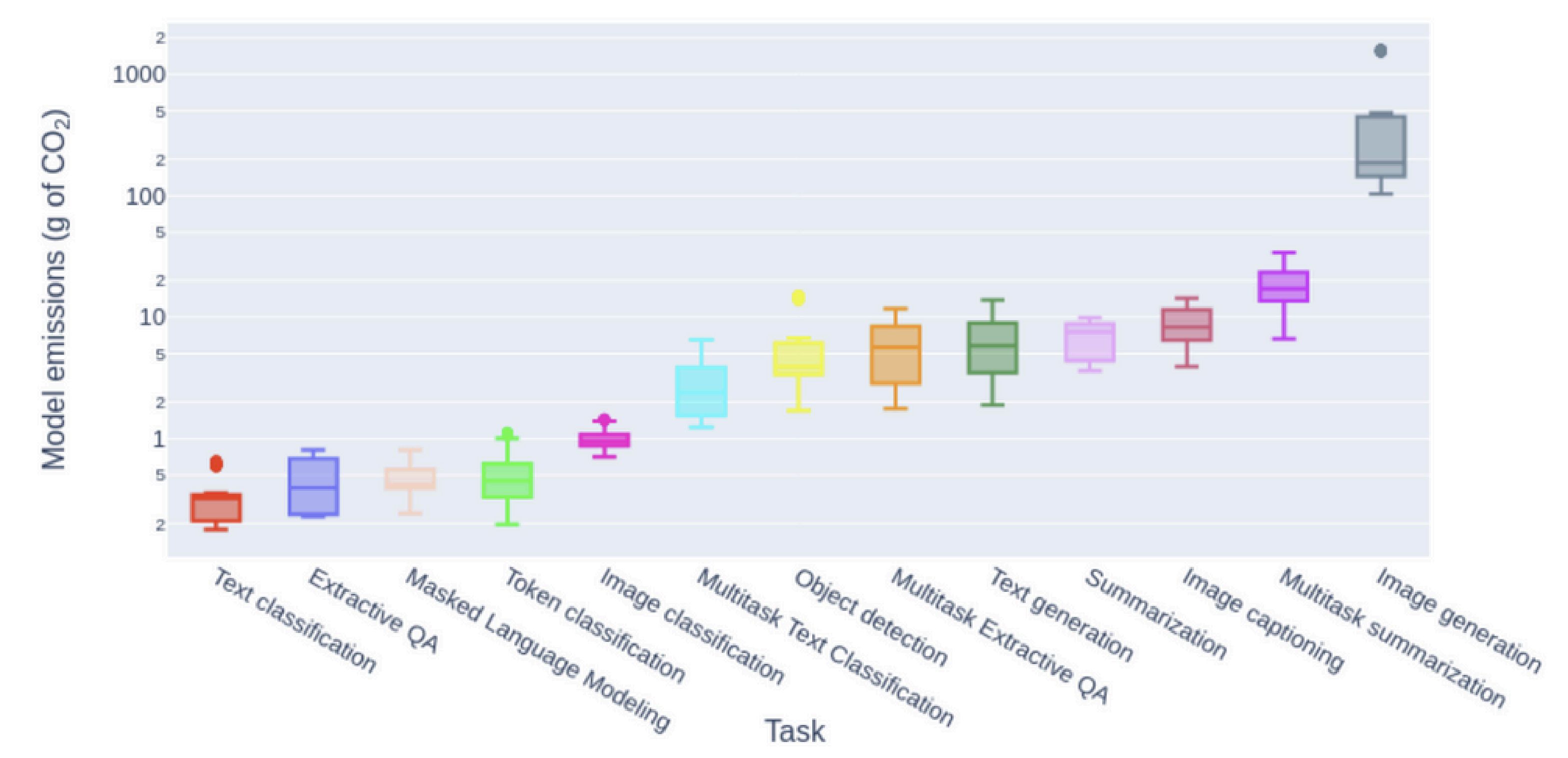
Exemplary Equivalents

4.46 % of weekly American household emissions

18 miles driven

3 days of 32-inch LCD TV watched

HOW TO REDUCE EMISSIONS ?



01 Reduce the Time and Space Complexity

02 Appropriate Data Structures: Using data structures that best meet specific needs (e.g., using dynamic arrays or hash tables)

03 Code optimization: Inlining, avoiding redundancies

04 Memory management: optimizing allocation and deallocation, using cache

05 Parallelization optimization

For Machine Learning :

06 Changing the task

07 Prune your model : structured pruning

08 Using adapters instead of fine tuning

09 Changing the type of architecture : CNN, RNN, MLP...

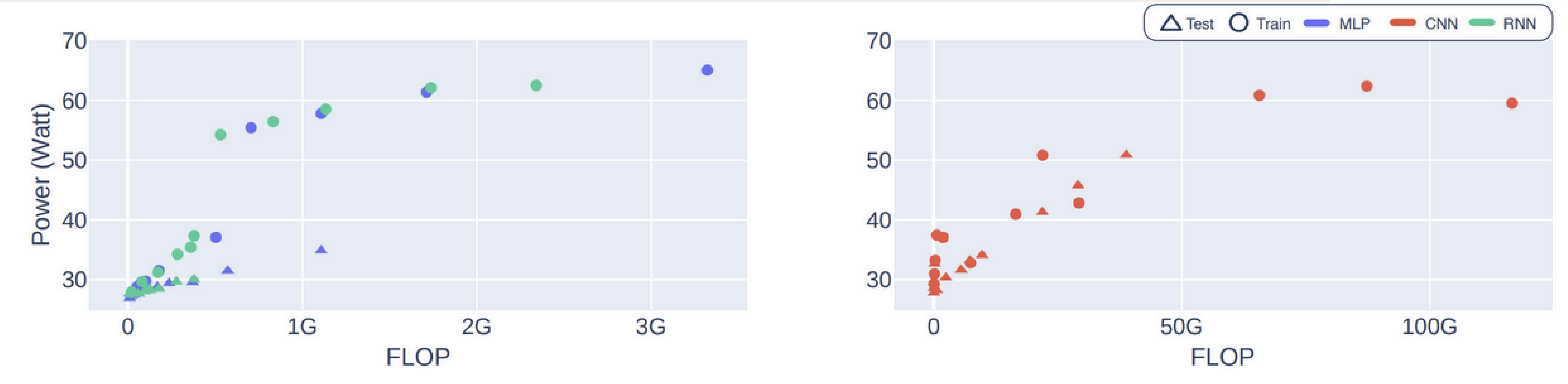


Figure 2: *Average power during training (circles) and test (triangles) in function of FLOP. The architectures are split with MLP/RNN on the left and CNN on the right.*