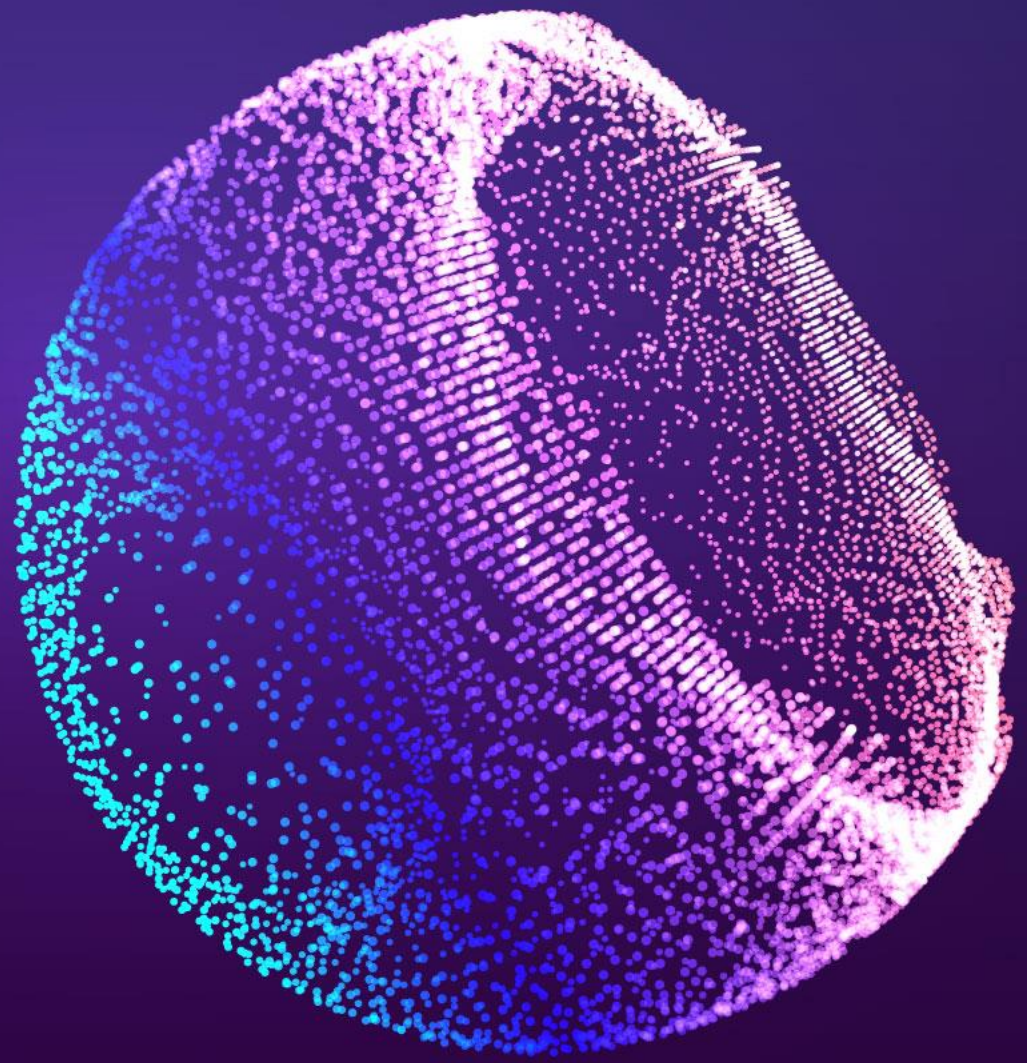
Accenture Labs

Inria

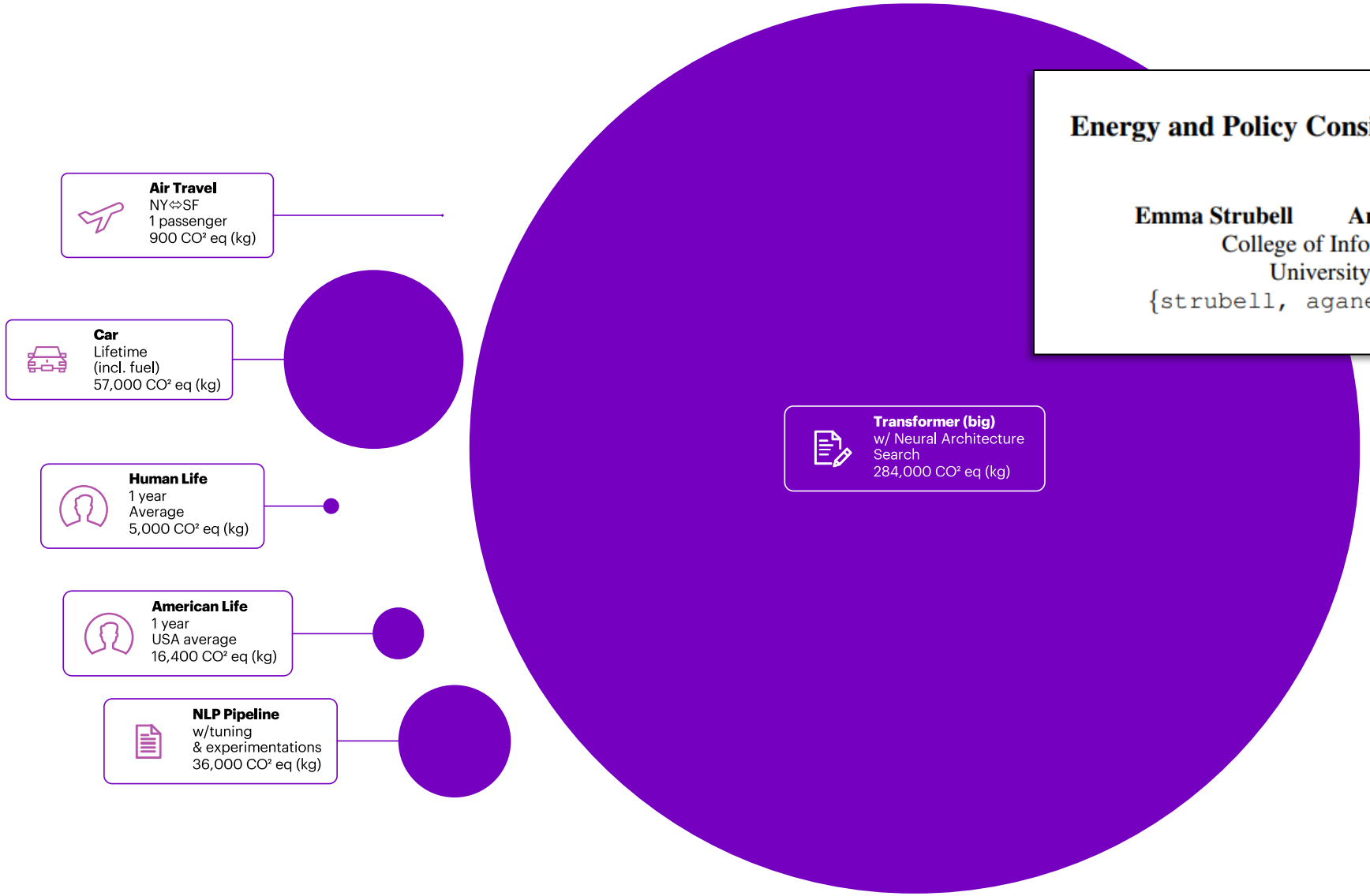
How can we evaluate the energy consumption of Machine Learning?

JRAF2023

Charlotte Rodriguez



Artificial Intelligence sustainability

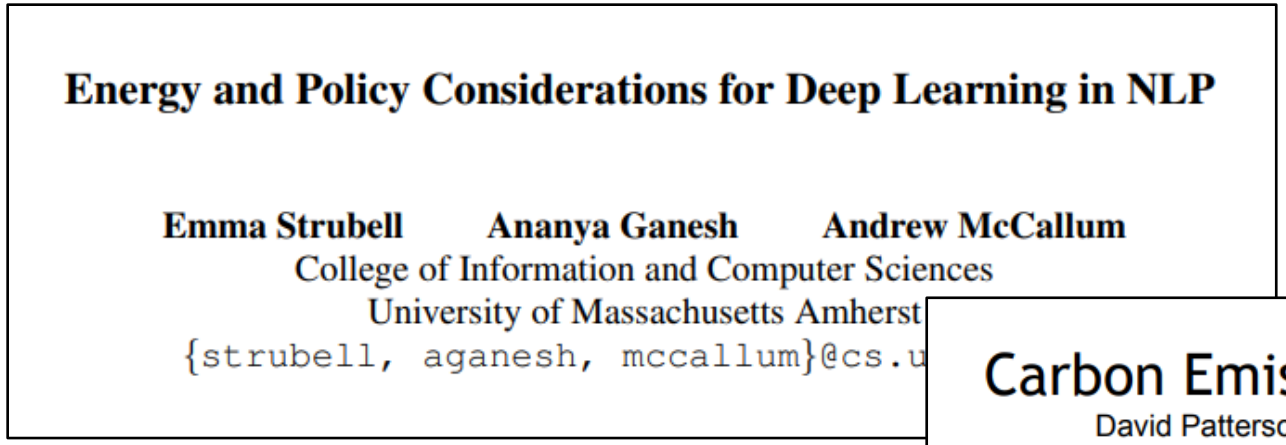


Energy and Policy Considerations for Deep Learning in NLP

Emma Strubell **Ananya Ganesh** **Andrew McCallum**
College of Information and Computer Sciences
University of Massachusetts Amherst
{strubell, aganesh, mccallum}@cs.umass.edu

Graph source: Energy and Policy Considerations for Deep Learning in NLP, 57th Annual Meeting of the Association for Computational Linguistics (ACL)

No consensus on how to evaluate the consumption



Energy and Policy Considerations for Deep Learning in NLP

Emma Strubell **Ananya Ganesh** **Andrew McCallum**
College of Information and Computer Sciences
University of Massachusetts Amherst
{strubell, aganesh, mccallum}@cs.u

versus

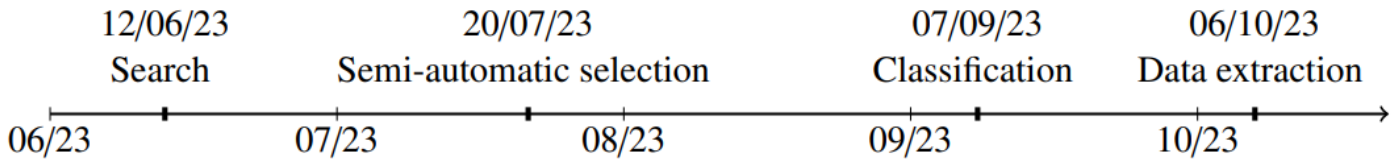
Carbon Emissions and Large Neural Network Training

David Patterson[1,2], Joseph Gonzalez[2], Quoc Le[1], Chen Liang[1], Lluis-Miquel Munguia[1],
Daniel Rothchild[2], David So[1], Maud Texier[1], and Jeff Dean[1]
{davidpatterson, qvl, crazydonkey, llmunguia, davidso, maudt, jeff}@google.com,
{pattrsn, jegonzal, drothchild}@berkeley.edu

"[the author's] energy estimate for NAS ended up 18.7X too high for the average organization (...) and 88X off in emissions for energy-efficient organizations like Google".

Systematic Literature Review (SLR)

Question: What tools and methods (not shaped into tools) currently permit to evaluate the energy consumption of machine learning computing tasks?



Data sources: ACM Digital Library, IEEE Xplore Digital Library, Google Scholar search engine (GS)

GS presents several constraints:

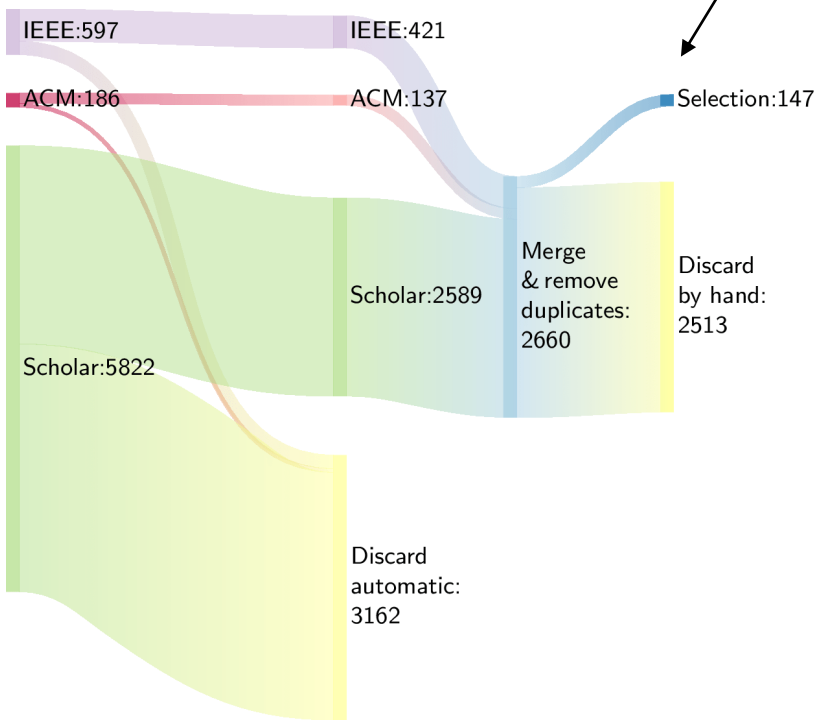
- search query may contain at most 256 characters.
- 1 query will provide at most 1000 results, even if it has more associated results (actual number of results is displayed by GS, even though the user does not have access to all of them).
- GS official interface only permits the user to save results to the user's Google Scholar Library by selecting the star icon, and then exporting said library. GS will tend to block any behavior deviating from this usage mode.

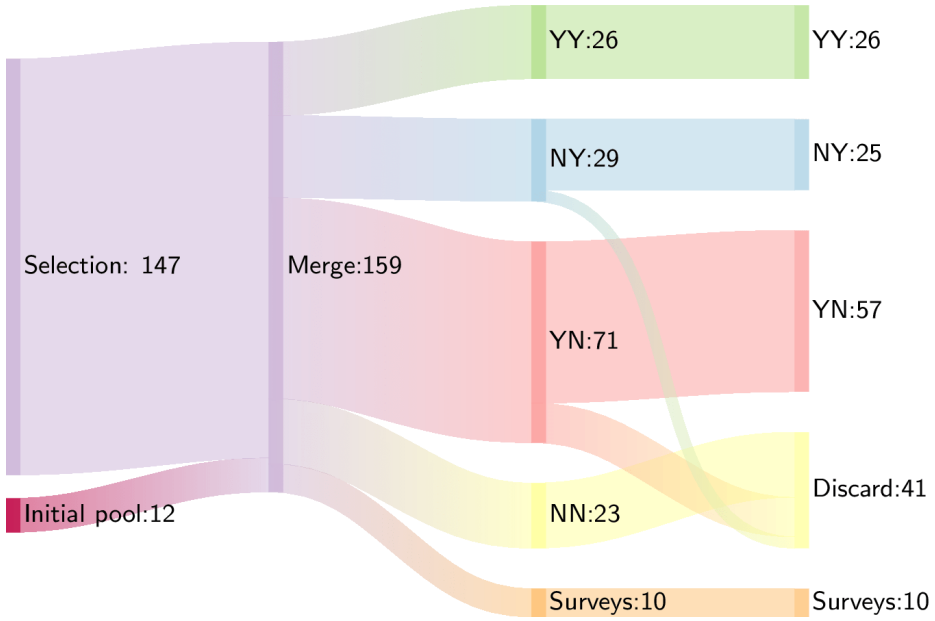
SLR: overview of the search steps

Initial pool of 25 primary and secondary studies +

Search steps:

1. Search by key-words (based on title)
2. Semi-automated selection (based on title)
3. Selection by hand (based on abstract and full text)
4. Classification and subsequent final selection (based on abstract and full text)

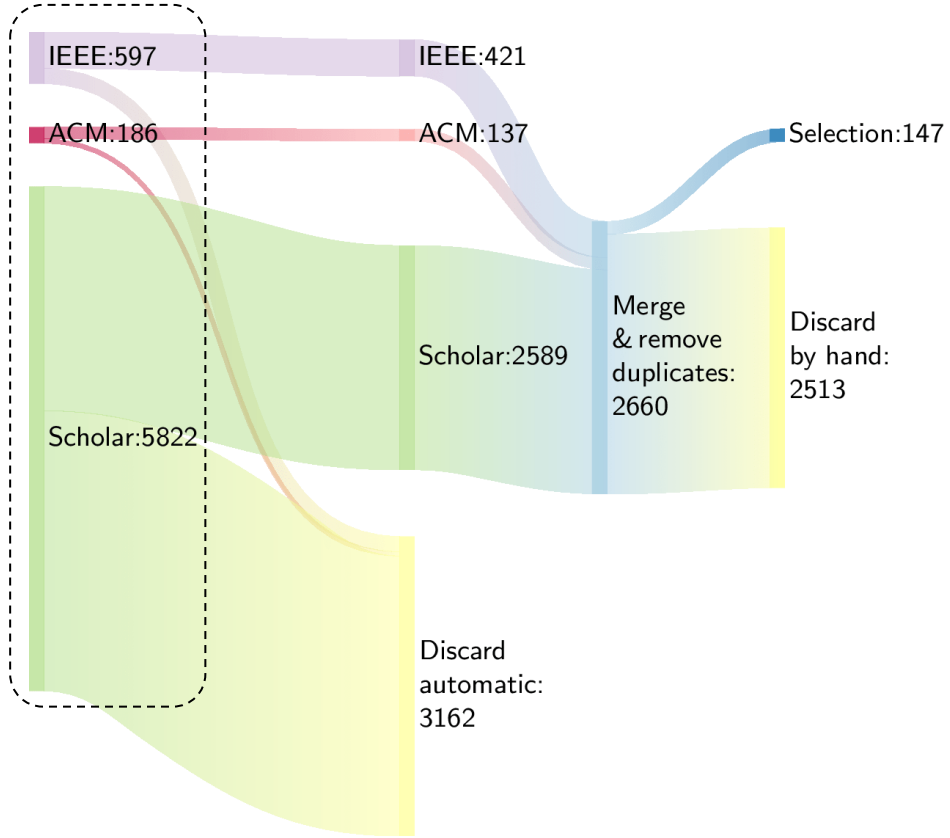




SLR: search step 1

Search steps:

1. **Search by key-words (based on title)**
2. Semi-automated selection (based on title)
3. Selection by hand (based on abstract and full text)
4. Classification and subsequent final selection (based on abstract and full text)



Key-words:

i. machine learning, deep learning, computing, information and communications technology, ICT, artificial intelligence, AI, natural language processing, NLP, neural network, neural networks, CNN, DNN, computation, computations, software, process-level, server, virtual machine, federated learning, distributed learning

ii. measure, measuring, estimate, estimation, consumed, consumption, predict, prediction, predicting, track, tracking, report, reports, reporting, account, quantify, quantifying, monitor, monitoring, evaluate, evaluating

iii. energy, power, environmental impact, carbon footprint, carbon emissions, carbon impact.

Simultaneously excluded all results containing any of the following keywords: wind, building, buildings, vehicles, homes, ships, solar, photovoltaic, vehicle.

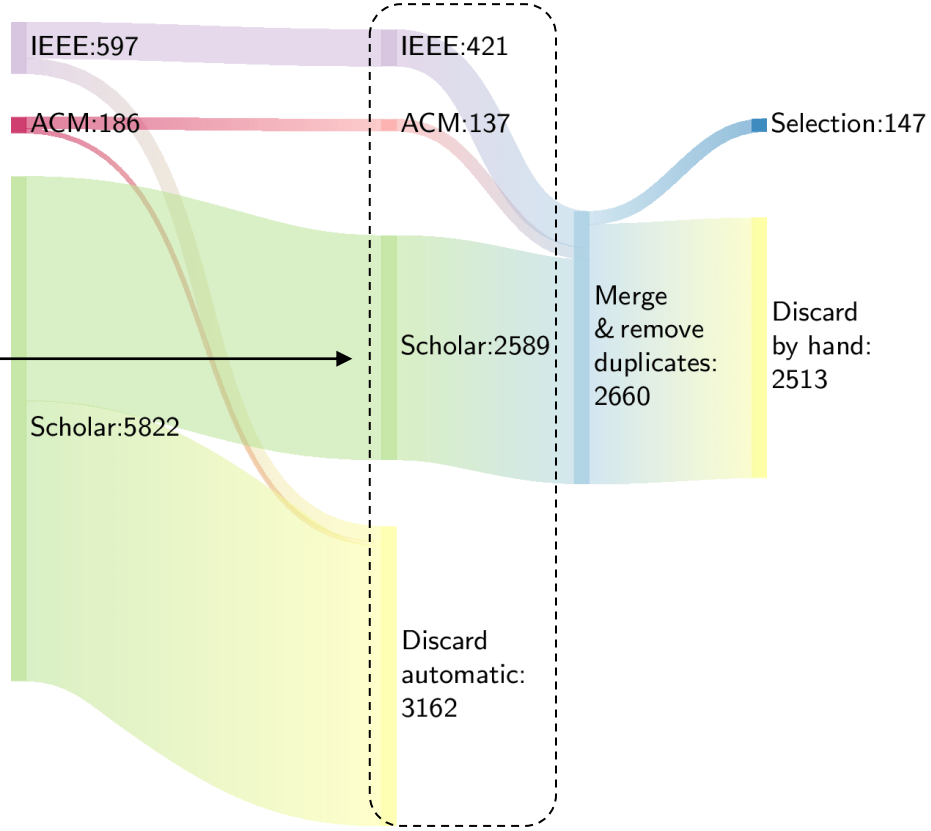
Remarks:

- Use the operators AND, OR, NOT, parenthesis and “phrase word for word” to build appropriate **queries** for each of the data sources.
- For GS, the original query had to be divided into **103 sub-queries to meet the constraints of GS**. The results of these sub-queries where then merged together, the duplicates removed.

SLR: search step 2

Search steps:

1. Search by key-words (based on title)
2. **Semi-automated selection (based on title)**
3. Selection by hand (based on abstract and full text)
4. Classification and subsequent final selection (based on abstract and full text)



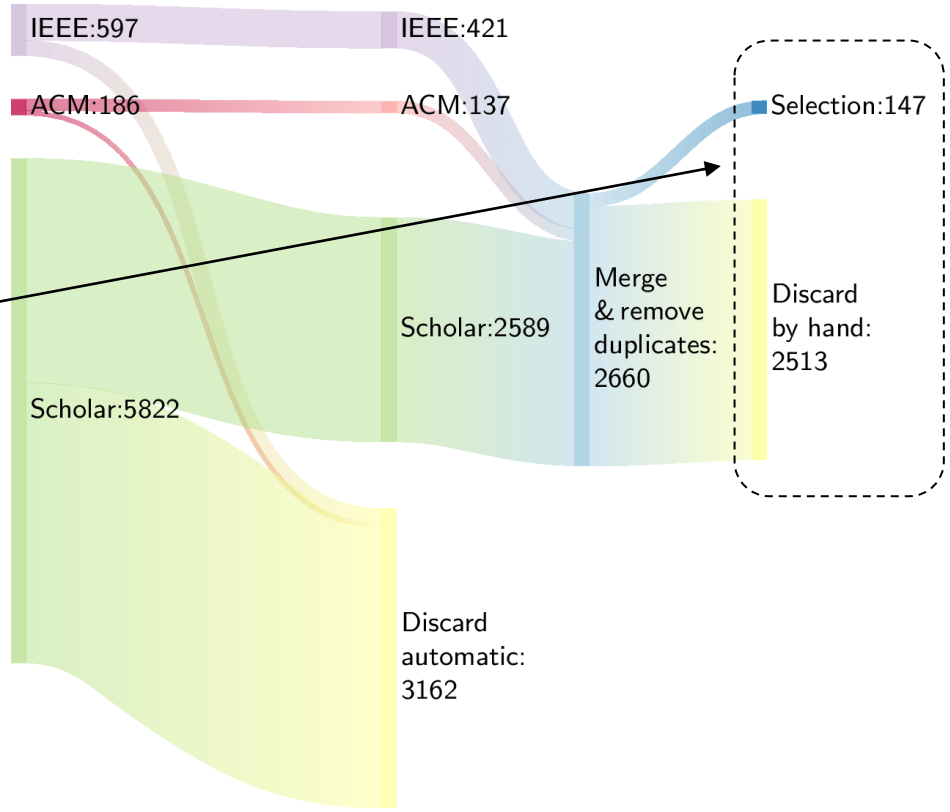
Remarks:

- First part of the selection process
- Based on the results' titles and on the relevance to the research question only.
- Consists in identifying words (among all words contained in the results' titles) that rule a title containing any of these words, as **off-topic**.

SLR: search step 3

Search steps:

1. Search by key-words (based on title)
2. Semi-automated selection (based on title)
3. **Selection by hand (based on abstract and full text)**
4. Classification and subsequent final selection (based on abstract and full text)



Remarks:

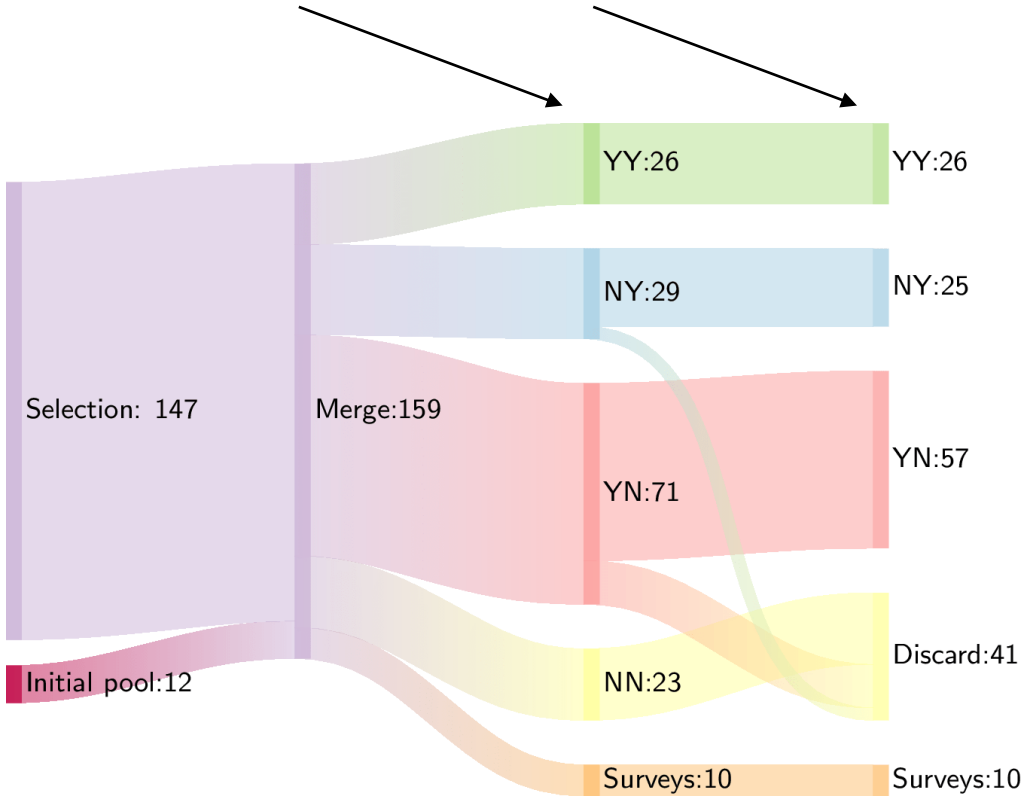
- Second part of the selection process
- Based on assessors reading the results' titles, abstracts and full text if necessary.
- Three assessors share this task.

SLR: search step 4

Search steps:

1. Search by key-words (based on title)
2. Semi-automated selection (based on title)
3. Selection by hand (based on abstract and full text)
4. **Classification and subsequent final selection (based on abstract and full text)**

creation \ for ML	yes	no
yes	YY	YN
no	NY	NN
survey	SY	SN



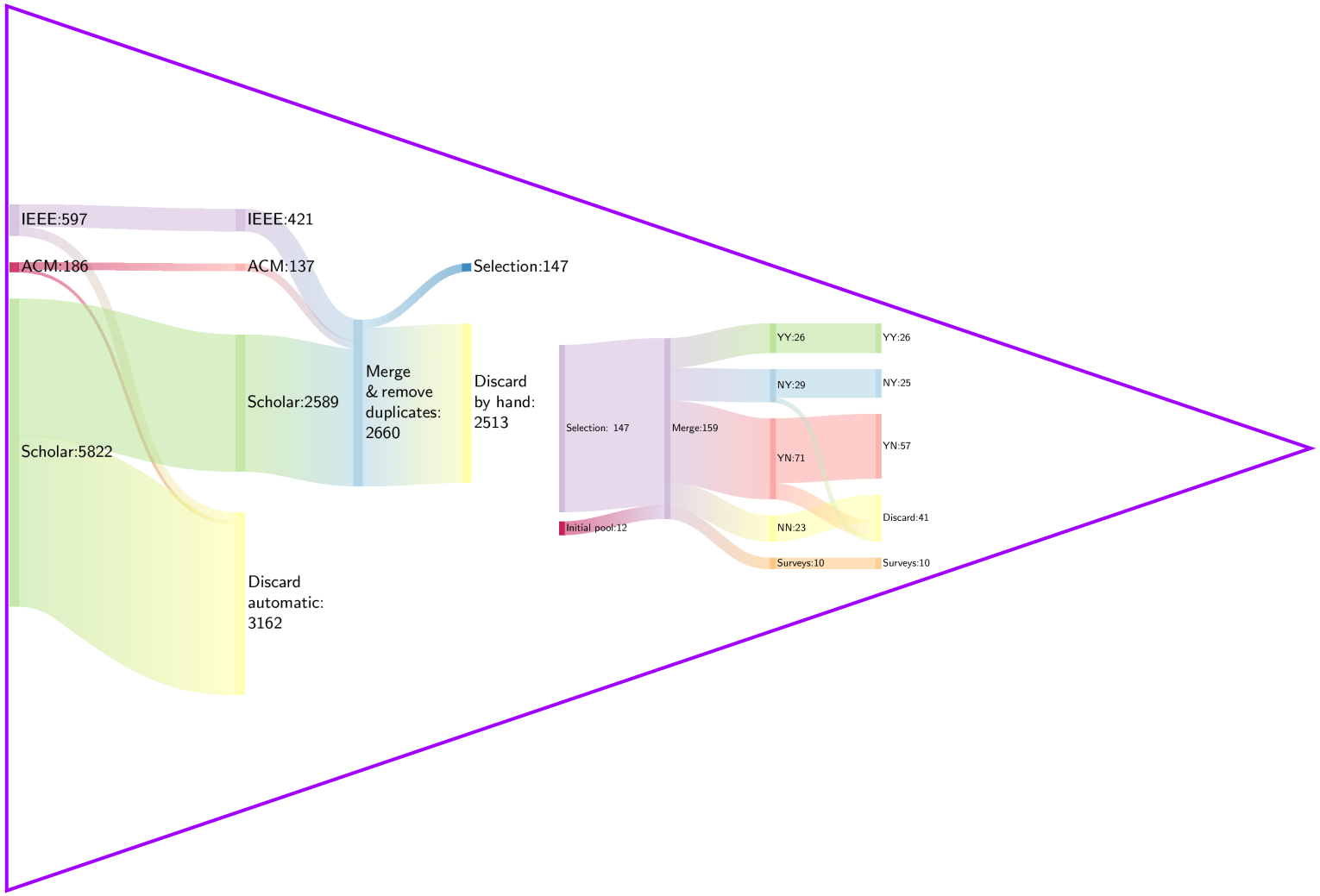
Classify the results selected in Step 3 according to two criteria:

I. If primary study, has a tool or method been created by the authors or not (= used only)? Otherwise, the result is a secondary study (survey). Three possible values: “Yes – creation”, “No – no creation”, “Survey”.

II. Is the result specifically concerned with AI applications or not (= concerned with software in general, virtual machines, data centers, etc.)? Two possible values: “Yes – for ML”, “No – not for ML”.

SLR: search steps summary

6605
+ 12
results



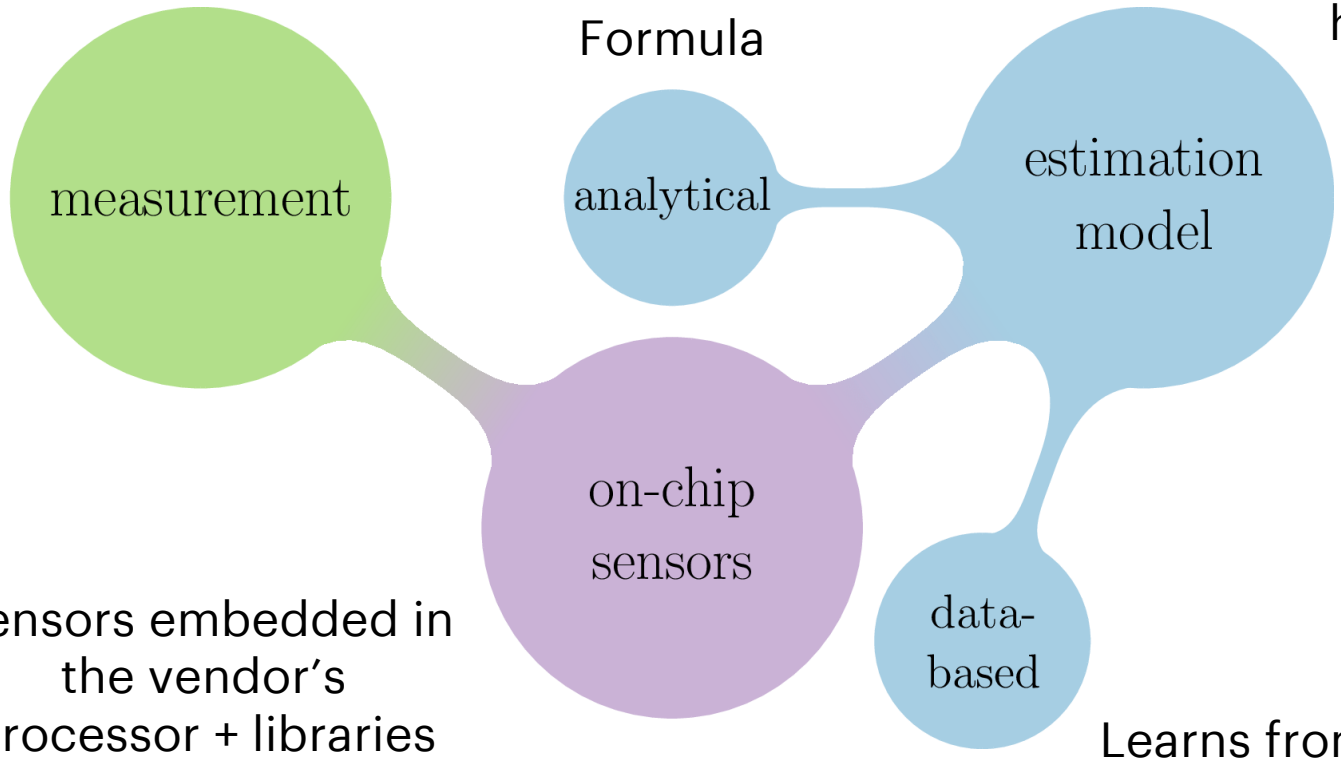
118
reviewed
results

Different approaches to evaluate energy consumption

Actual measurements of current intensity, etc.

Formula

Estimation from indirect evidence (activity of hardware, charact. of target application)



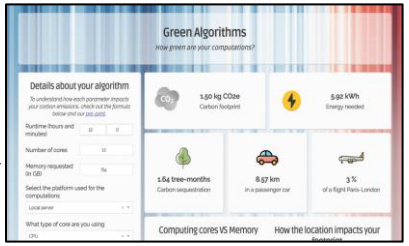
Sensors embedded in the vendor's processor + libraries

Learns from data

Examples of tools and methods

External Power Meter

ECO2AI



GREEN ALGORITHMS

measurement

analytical

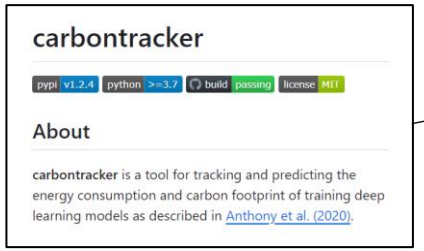
estimation model

FLOPS

FLOPS

on-chip sensors

data-based



CARBON TRACKER

CODE CARBON

Comparison framework

Qualitative study

- On the method or tool (approach, output, …)
- On the usage (hardware, software constraints, …)
- On the surrounding project (availability, context, …)

Quantitative study

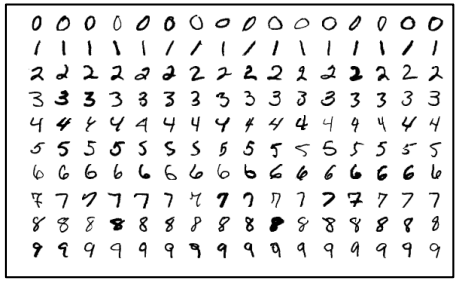
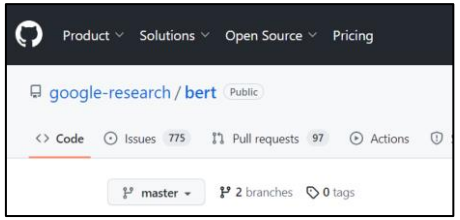


Image Classifier on MNIST
Script: PyTorch example Basic MNIST Example
GitHub: pytorch/examples/tree/main/mnist

Image Classifier on CIFAR10
Script: PyTorch tutorial "Training a classifier"
Pytorch website at: tutorials/beginner/blitz/

Resnet18 on ImageNet
Script: recipe for training Resnet18 from PyTorch
GitHub: pytorch/vision/references/classification/



Bert-base on SQUADv1.1
Script: recipe for fine-tuning Bert from google-research
GitHub: google-research/bert/

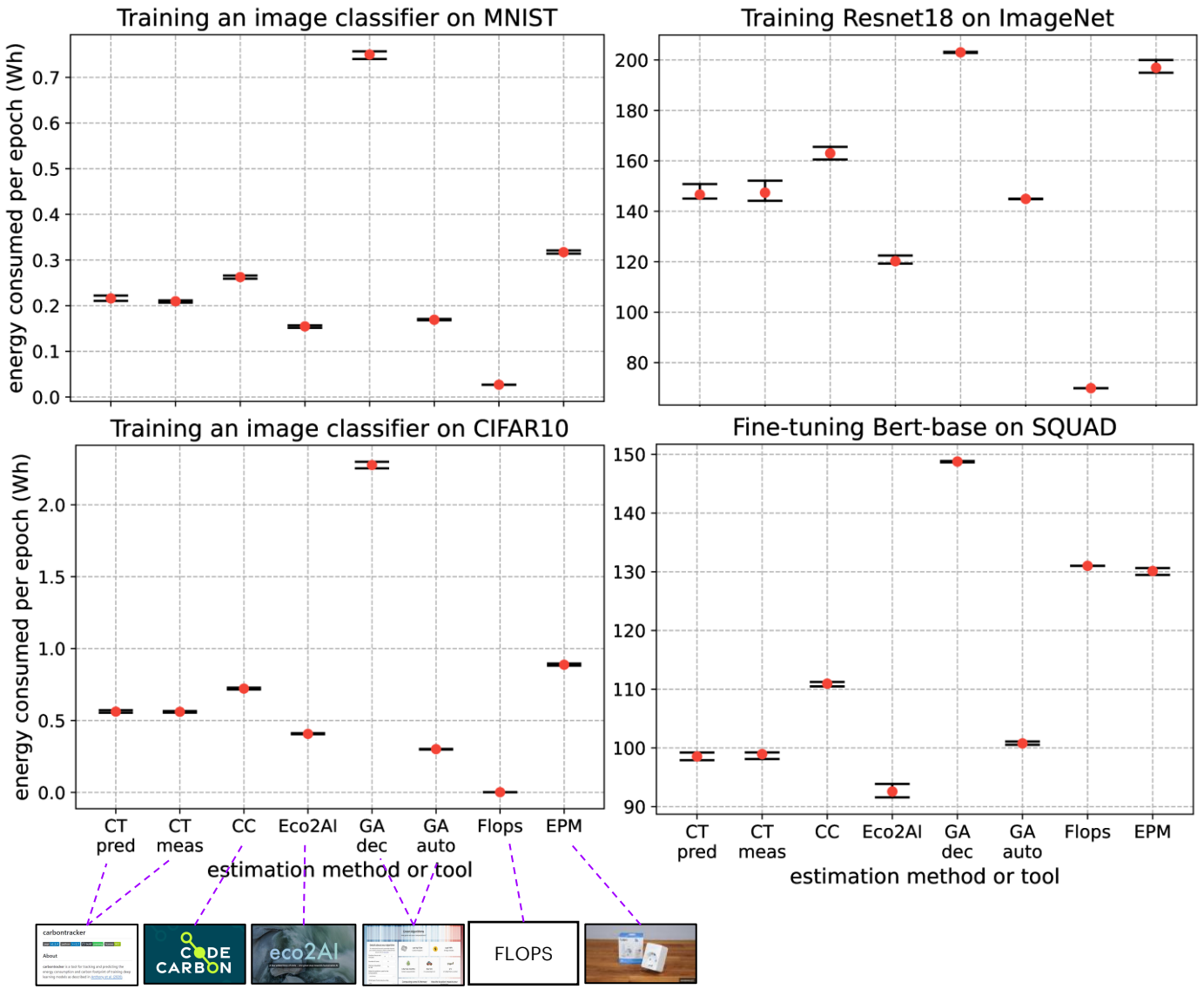
Comparison framework: application

Evaluation tools and methods tested:
CT:pred, CT:meas, CC, Eco2AI, GA:dec, GA:auto, Flops, EPM.

- Some tools are tested in different modes
  - GA:dec (100% hardware usage), GA:auto (monitoring + average of hardware usage)
  - CT:meas (normal mode), CT:pred (prediction based on 1 epoch)
- EPM = external power meter

Observations:

- Different outputs for different approaches, though some remain comparable
- Different outputs for some tools and methods, depending on the ML task



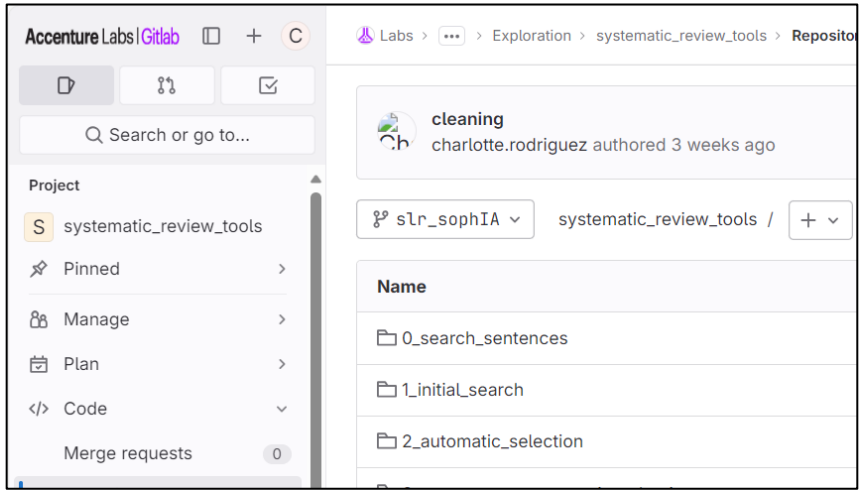
Conclusion

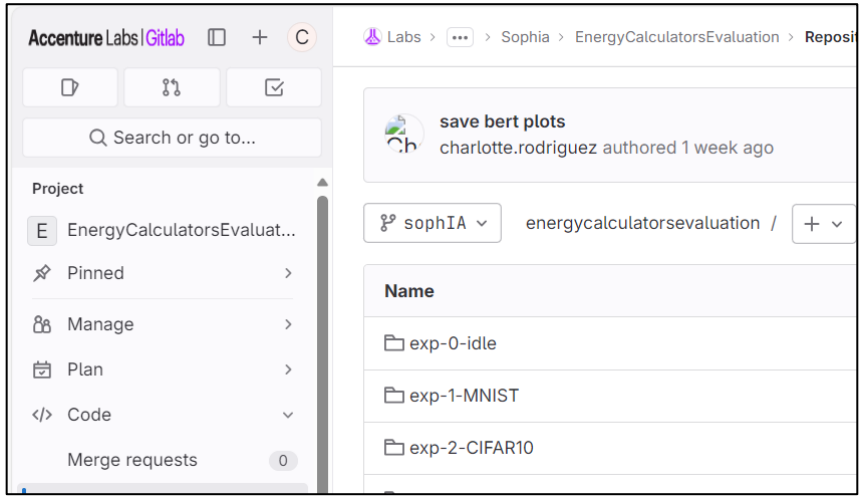
Limits:

- SLR: search done in June
- SLR: no backward or forward search
- Experiments: no baseline

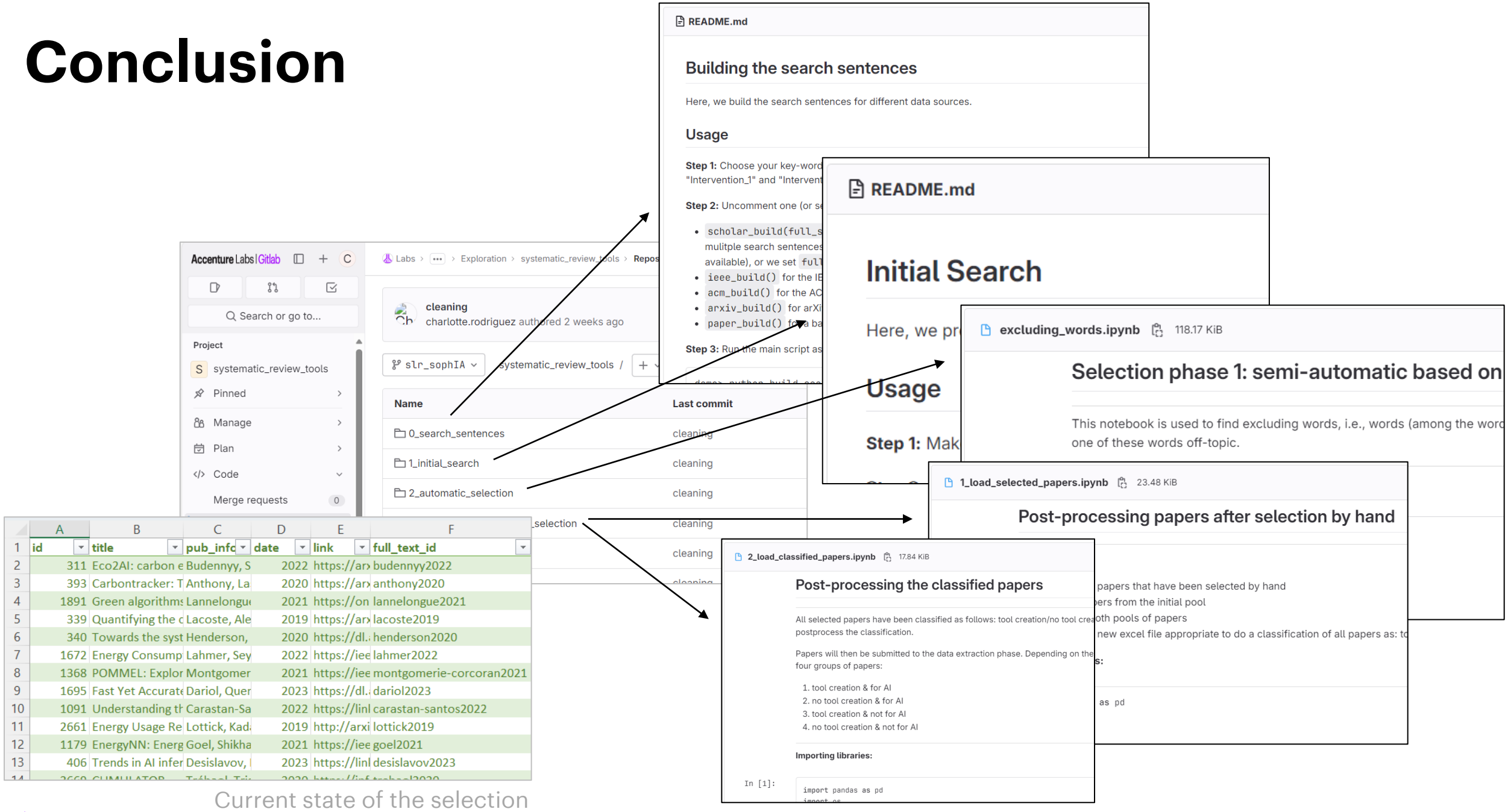
Next:

- This study on arxiv and github (SLR & exp.)
- Work on some of the limits
- Recommendations
- Scheduling tool for decentralized learning





Conclusion

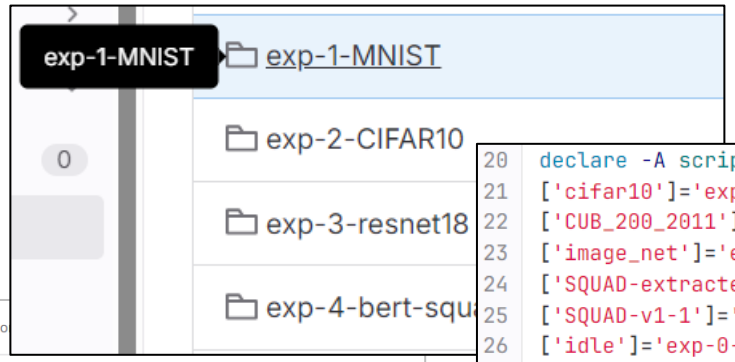


	A	B	C	D	E	F
1	id	title	pub_info	date	link	full_text_id
2	311	Eco2AI: carbon e	Budennyy, S	2022	https://arx	budennyy2022
3	393	Carbontracker: T	Anthony, La	2020	https://arx	anthony2020
4	1891	Green algorithms	Lannelongue	2021	https://on	lannelongue2021
5	339	Quantifying the c	Lacoste, Ale	2019	https://arx	lacoste2019
6	340	Towards the syst	Henderson,	2020	https://dl.	henderson2020
7	1672	Energy Consump	Lahmer, Sey	2022	https://iee	lahmer2022
8	1368	POMMEL: Explor	Montgomer	2021	https://iee	montgomerie-corcoran2021
9	1695	Fast Yet Accurate	Dariol, Quer	2023	https://dl.	dariol2023
10	1091	Understanding th	Carastan-Sa	2022	https://lin	carastan-santos2022
11	2661	Energy Usage Re	Lottick, Kad	2019	http://arxi	lottick2019
12	1179	EnergyNN: Energ	Goel, Shikha	2021	https://iee	goel2021
13	406	Trends in AI infer	Desislavov,	2023	https://lin	desislavov2023

Current state of the selection

Conclusion

Add a new ML computing task script + additional modifications:

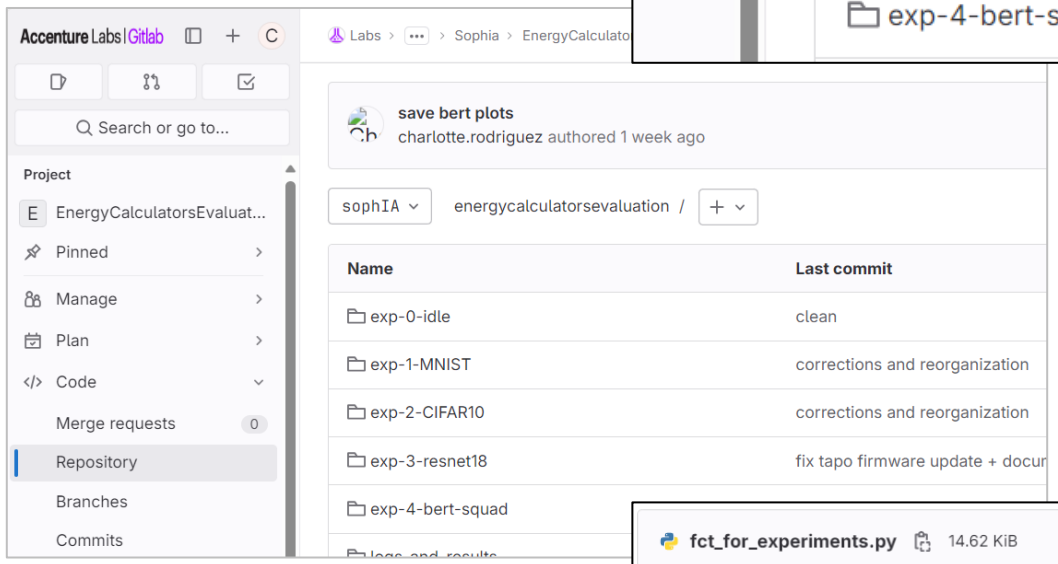



```
declare -A scripts=( ['mnist'
['cifar10']='exp-2-CIFAR10/CI
['CUB_200_2011']='exp-3-resne
['image_net']='exp-3-resnet18
['SQUAD-extracted']='exp-4-be
['SQUAD-v1-1']='exp-4-bert-sq
['idle']='exp-0-idle/idle' )
```

Provide the path to the script

```
    if [[ $exp == 'mnist' ]]; then
        if [[ $dev_test == 'True' ]]; then
            opt7="--nb_batch_inferences=10"
            opt8="--epochs=2"
            opt="${opt} ${opt7} ${opt8}"
        fi
        if [[ $ml == 'training' ]]; then
            mkdir "${path_logs_and_results}/${i}/${exp}_model"
            opt9="--output_dir=${path_logs_and_results}/${i}/${exp}_model"
            opt="${opt} ${opt9}"
        fi
    fi
```

Specify options, ...



Specific preparation steps

```
def prepare_calculator(exp):
    #     Preparing the calculators     #

    tracker = None

    if exp.name_calc == 'code_ca
        output_file = exp.cc_out
        tracking_mode = 'machine
        # tracking_mode = 'proce
        if exp.online == True:
            measure_power_secs =
            tracker = EmissionsT
```

How to save the outputs, ...

fct_for_experiments.py 14.62 KiB

```
import sys
import os
_path = '.'
sys.path.append(os.path.join(_path))
import json
import time

# --- FOR CALCULATORS
from codecarbon import EmissionsTracker, OfflineEmissionsTracke
from carbontracker.tracker import CarbonTracker
```

fct_for_saving.py 14.67 KiB

```
import json
import os
import pandas as pd
from datetime import timedelta
import numpy as np
```

Or add a new evaluation tool or method + additional modifications:

Thank You

Comparison framework: detail

For each of the four ML computing tasks, we do the following experiment:

Repeat 5 times the following: for each evaluation tool or method that we want to test (in a random order), we record the duration of the computing task and energy consumed evaluation (output of the method or tool).

Computer	Alienware Desktop Computer
CPU	Intel Core i9-9900K CPU - 3.60GHz - with 8 cores (2 threads per core: 16 virtual cores) - TDP 95W
RAM	64,0 GB
Integrated GPU	Intel UHD Graphics 630
GPU 1	NVIDIA GeForce RTX 2080 SUPER - TDP 250W
GPU 2	NVIDIA GeForce RTX 2080 SUPER - TDP 250W
OS Version	Ubuntu 22.04.1 LTS (Jammy Jellyfish)
Python	3.10.6