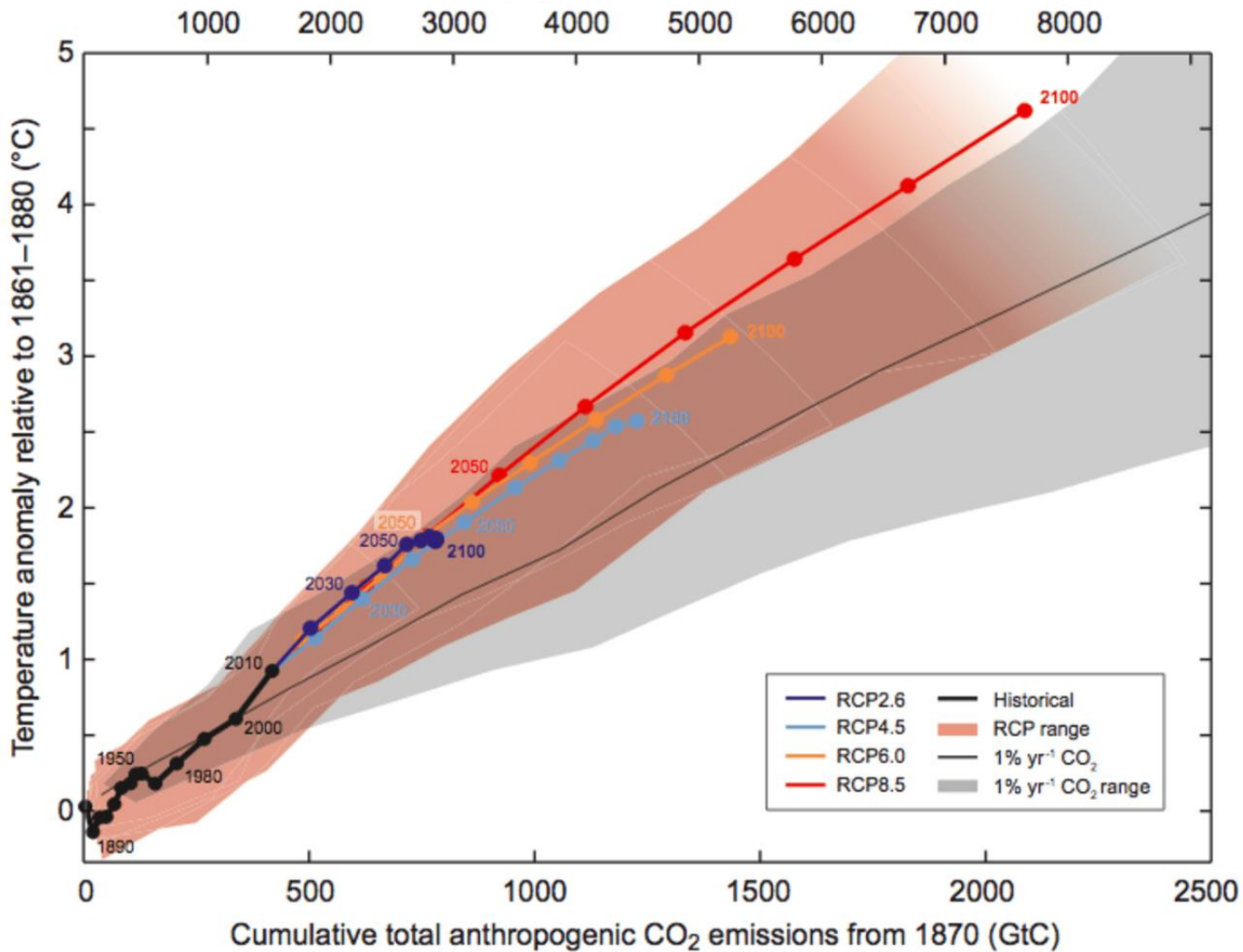


Session #28: Environmental Impact of Self-Supervised Models

Thursday, October 18
CSCI 601.771: Self-supervised Statistical Models





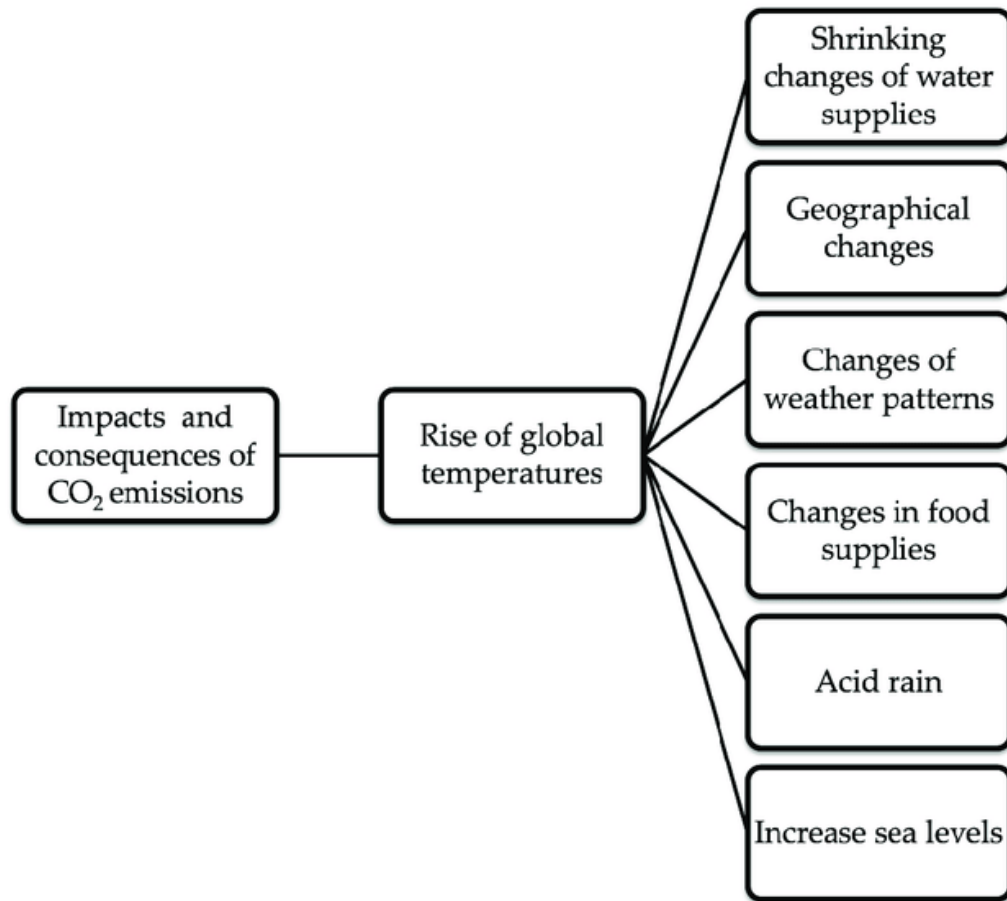
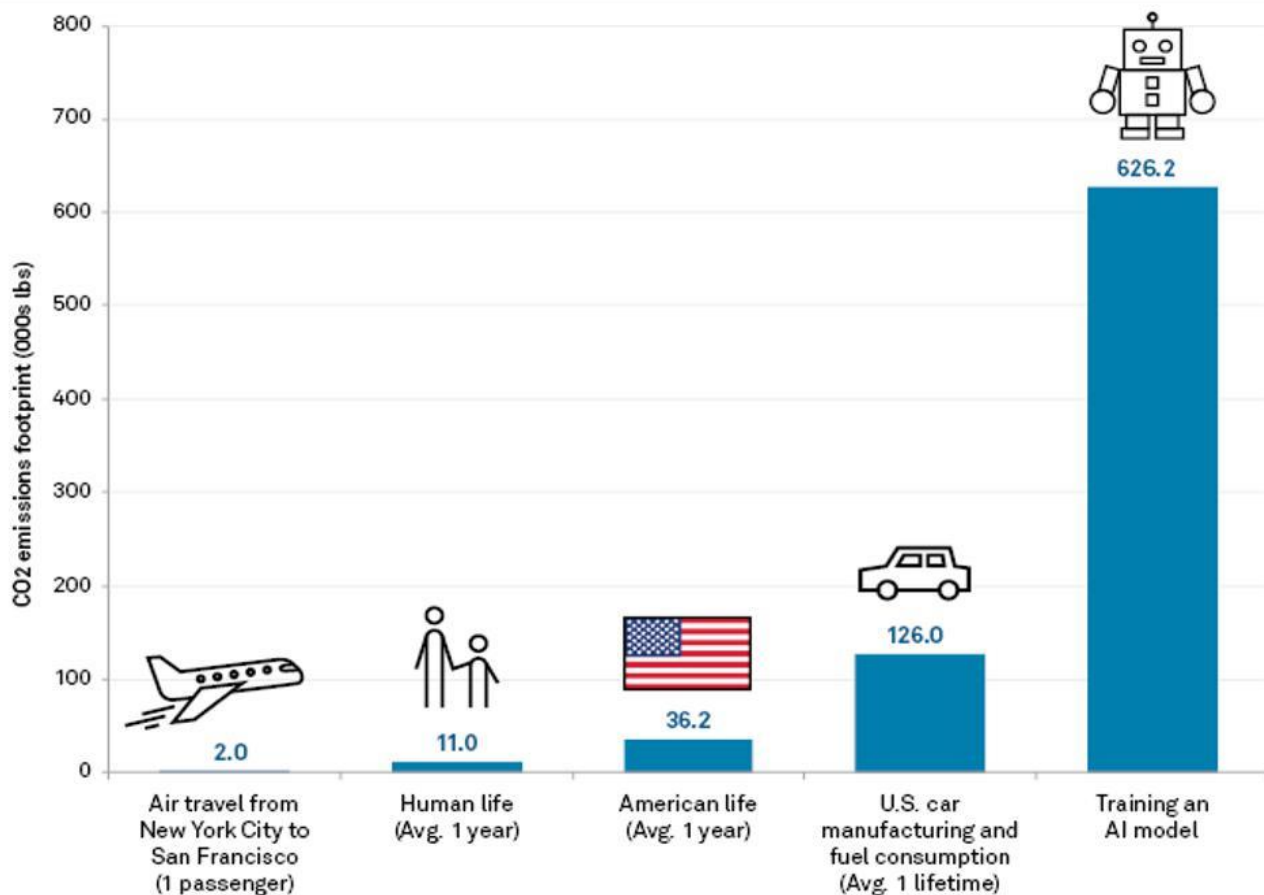


Figure 2. Impacts and consequences of CO₂ emissions on the environment.

CO2 emission benchmarks



Data compiled Oct. 9, 2019.

An "American life" has a larger carbon footprint than a "Human life" because the U.S. is widely regarded as one of the top carbon dioxide emitters in the world.

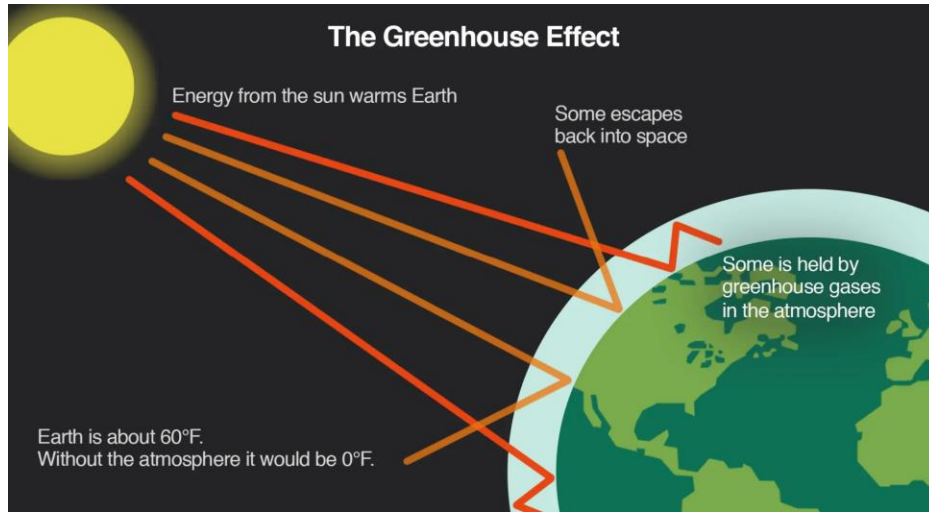
Source: College of Information and Computer Sciences at University of Massachusetts Amherst

Stakeholders

Ammar, Fadil, Karan

BACKGROUND

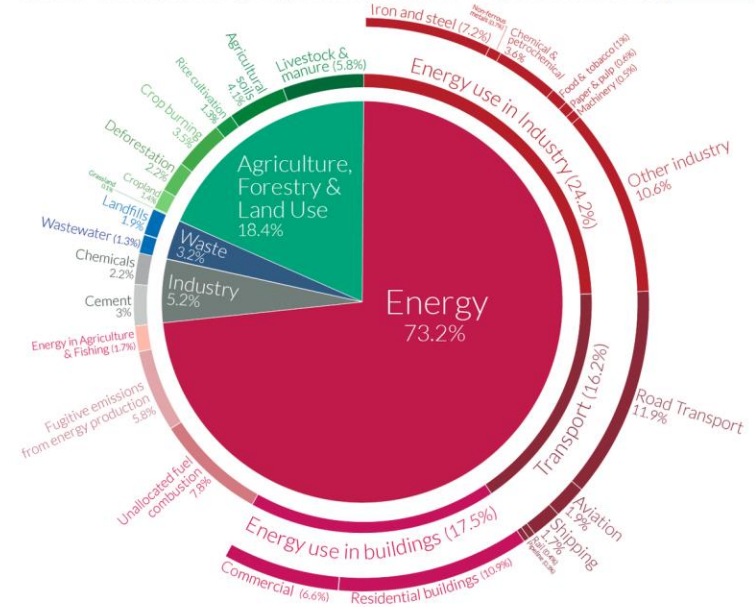
- CO₂ (and other types of greenhouse gases, such as methane and ozone) contributes to the greenhouse effect by trapping the heat from the sun within the atmosphere without letting it dissipate.



<https://www.climatecentral.org/climate-matters/the-greenhouse-effect>

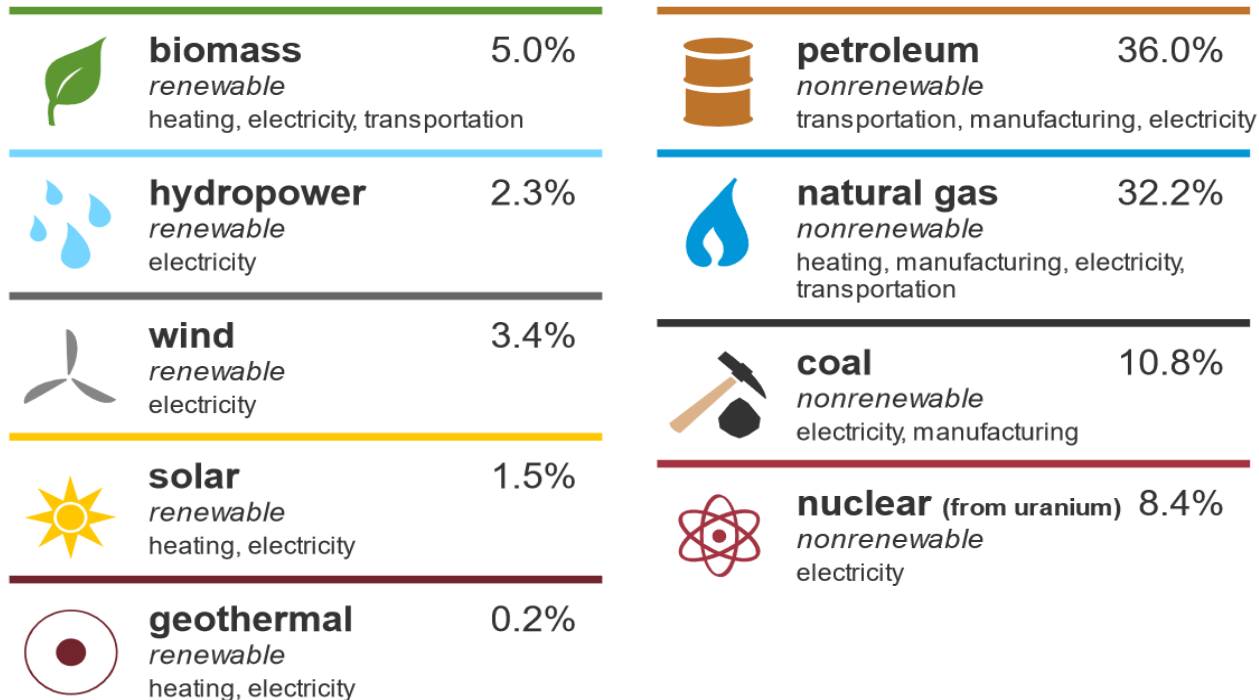
Global greenhouse gas emissions by sector
This is shown for the year 2016 – global greenhouse gas emissions were 49.4 billion tonnes CO₂eq.

Our World in Data



BACKGROUND

U.S. energy consumption by source, 2021



BACKGROUND

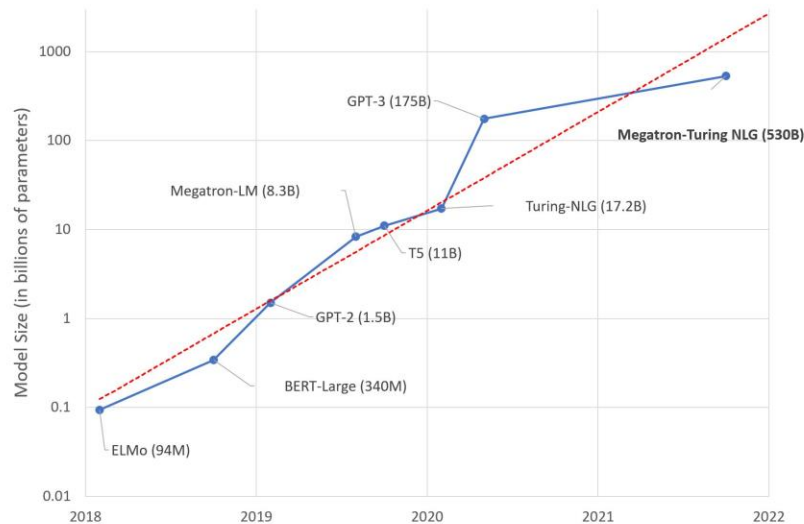
- In 2018, it was estimated that global data center energy use represented close to 1% of global energy usage

	Electricity net consumption (billion kWh)
World	22,027
Portugal	49
Hong Kong	45
Denmark	34
Morocco	31
New Zealand	40

US Energy Information Administration: 2018 Electricity consumption by country

BACKGROUND

- Model Size growing exponentially!!
- Strubell et al showed Transformer training emissions comparable to 5 cars!!
- And it will take 1200 trees to get rid of the emissions!!!



COMPUTING CO2 INTENSITY

Software Carbon Intensity (SCI) =
carbon emissions per one
functional unit R

$$SCI = ((E * I) + M) \text{ per } R$$

Energy (E) consumed by a
software (KWh)

R = one machine learning job

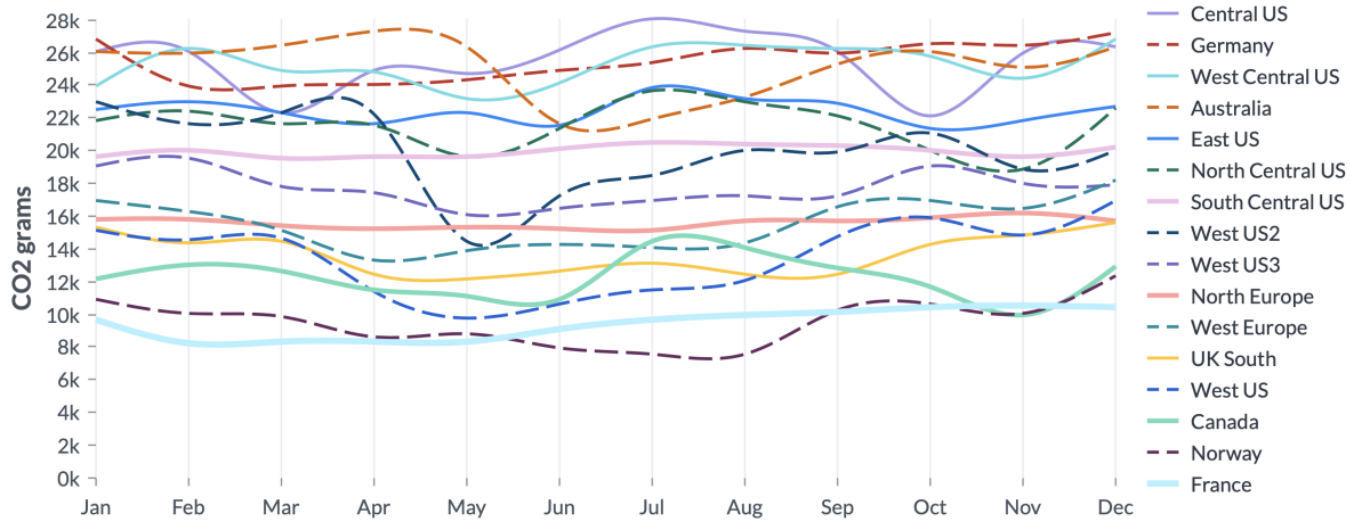
Operational Emission (O) [used
in the article]

Location based marginal carbon
emission (gCO₂eq/KWh)

$$SCI = (O + M) \text{ per } R$$

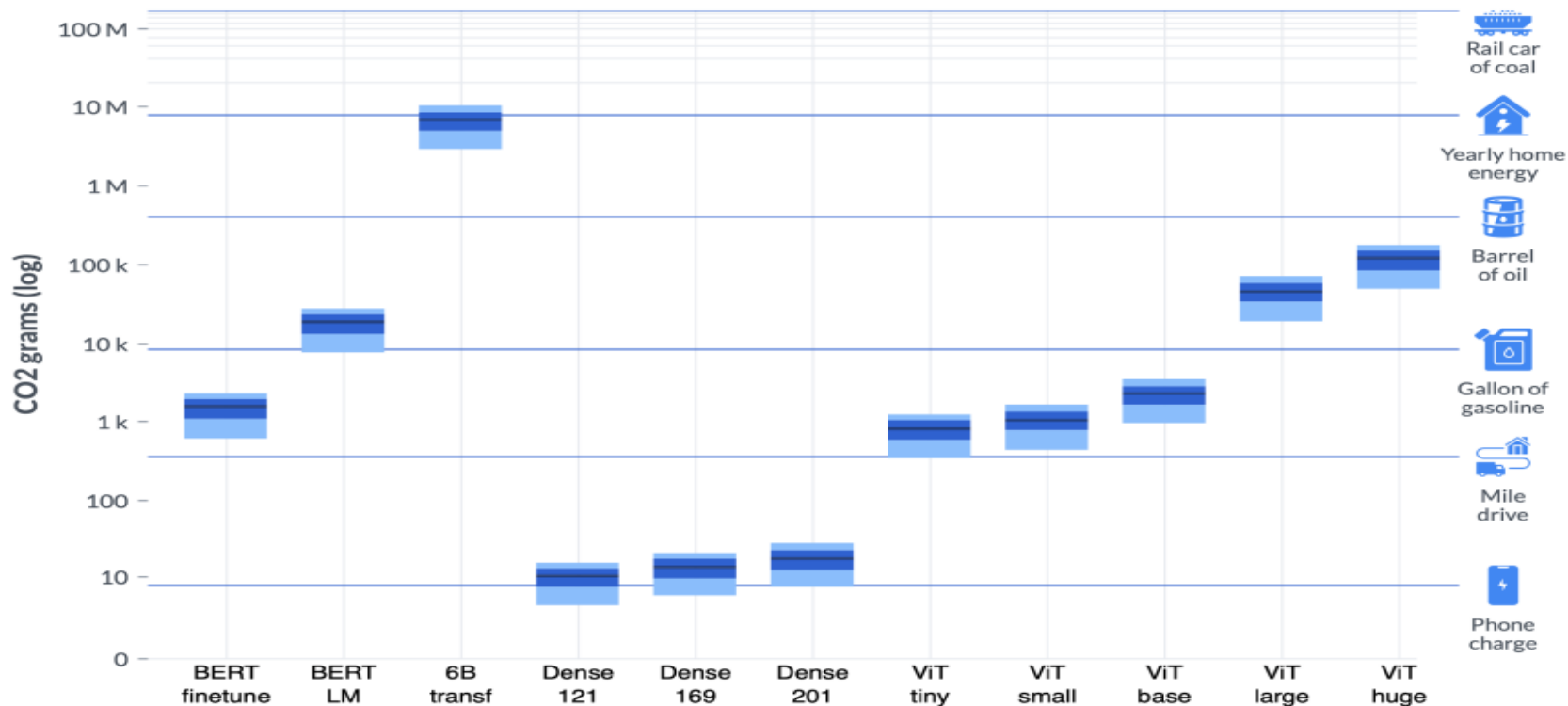
Carbon emission by hardware
[ignored in the article]

EMISSIONS BY REGION



- Each region has different sources of energy. Higher renewable source implies less carbon emissions.
- Training BERT (language modeling on 8 V100s for 36 hours).
- Each line is relatively flat.
- Large variation between the least carbon-intensive regions compared to the most carbon-intensive regions.

ELECTRICITY CONSUMPTION FOR AI WORKLOADS



GPU	4·V100	8·V100	256·A100	1·P40	1·P40	1·P40	1·V100	1·V100	1·V100	4·V100	4·V100
Hours	6	36	192	0.3	0.3	0.4	19	19	21	90	216

EMISSIONS BY TIME OF DAY

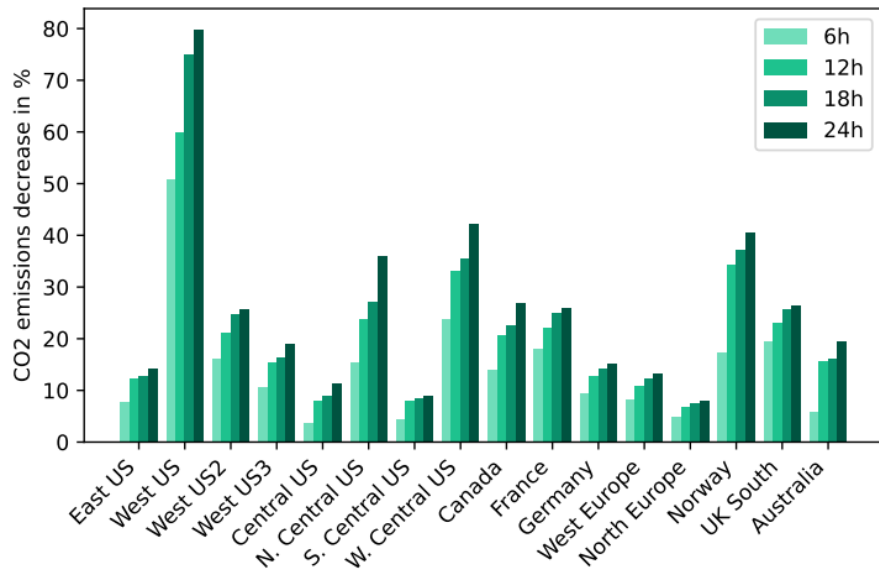
- The emissions vary by time of day because during the day, a region may have a higher mix of renewable energy or fossil-fuel based source.
- The amount of variation varies by region and time of year

BERT finetune	Central US	Hour	0:00	03:00	06:00	09:00	12:00	15:00	18:00	21:00
		Day 1	2,381	2,341	2,210	2,252	2,354	2,391	2,410	2,403
		Day 2	2,330	2,249	2,204	2,299	2,320	2,317	2,339	2,344
		Day 3	2,430	2,339	2,257	2,313	2,393	2,374	2,317	2,331

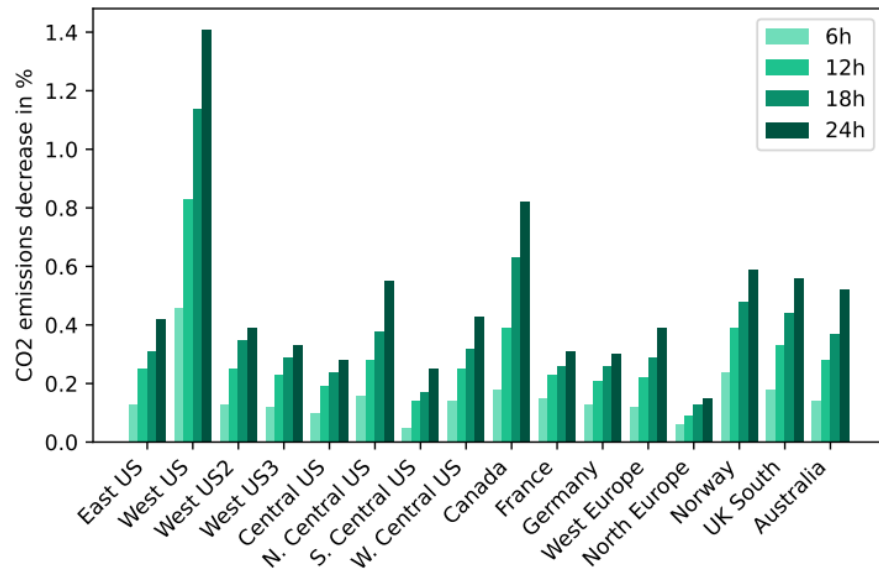
Finetuning the BERT-small model on a standard natural language inference task for around 6 hours on 4 NVIDIA V100 GPUs

OPTIMIZING CLOUD WORKLOADS

Flexible Start



(a) *Flexible Start* optimization for Dense 201.

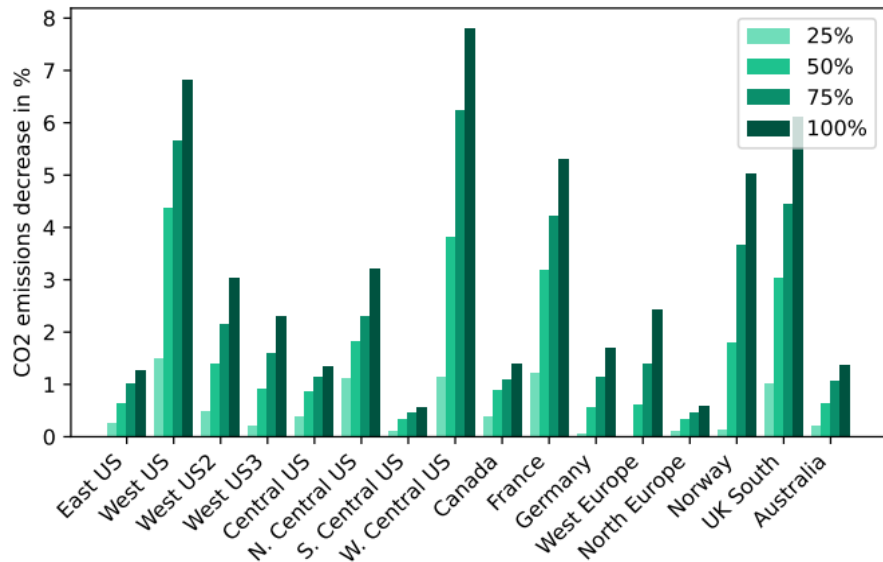


(b) *Flexible Start* optimization for 6B parameters Transformer.

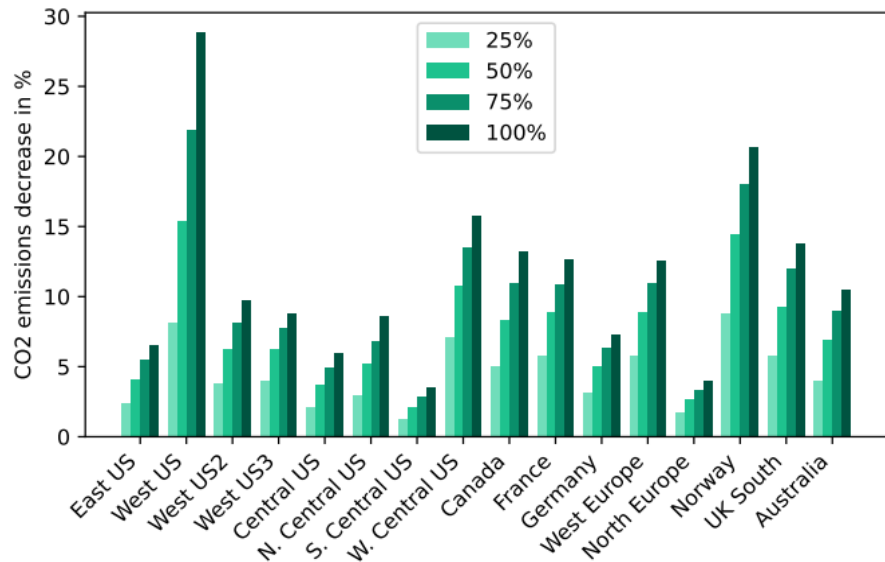
Proportion of emissions we expect to save if we change the start time by up to 24 hours.

OPTIMIZING CLOUD WORKLOADS

Pause and Resume



(a) *Pause and Resume* optimization for Dense 201.



(b) *Pause and Resume* optimization for 6B parameters Transformer.

Proportion of emissions we expect to save if we use "Pause and Resume" for a duration up to double the original duration.

OPTIMIZING CLOUD WORKLOADS

Comparison of the Two Methods

Model	BERT finetune	BERT LM	6B Transf.	Dense 121	Dense 169	Dense 201	ViT Tiny	ViT Small	ViT Base	ViT Large	ViT Huge
FS	14.5%	3.4%	0.5%	26.8%	26.4%	25.9%	5.6%	5.3%	4.2%	1.3%	0.5%
P&R	19.0%	8.5%	2.5%	27.7%	27.3%	27.1%	12.5%	12.3%	11.7%	4.7%	2.4%
Pauses / hr	0.23	0.3	0.15	0.06	0.07	0.08	0.3	0.3	0.3	0.23	0.14

Model	BERT finetune	BERT LM	6B Transf.	Dense 121	Dense 169	Dense 201	ViT Tiny	ViT Small	ViT Base	ViT Large	ViT Huge
FS	7.0%	4.1%	2.6%	1.8%	2.5%	2.7%	5.0%	4.8%	3.9%	3.3%	3.0%
P&R	9.5%	11.0%	11.4%	2.0%	2.8%	3.1%	11.0%	11.0%	10.8%	11.4%	11.3%
Pauses / hr	0.42	0.29	0.27	1.5	1.88	2.0	0.31	0.32	0.31	0.27	0.26

- Duration increased by 24 hours (top); duration increased by 100% (bottom).
- Allowed the same increase in duration, Pausing and Resuming (P&R) will

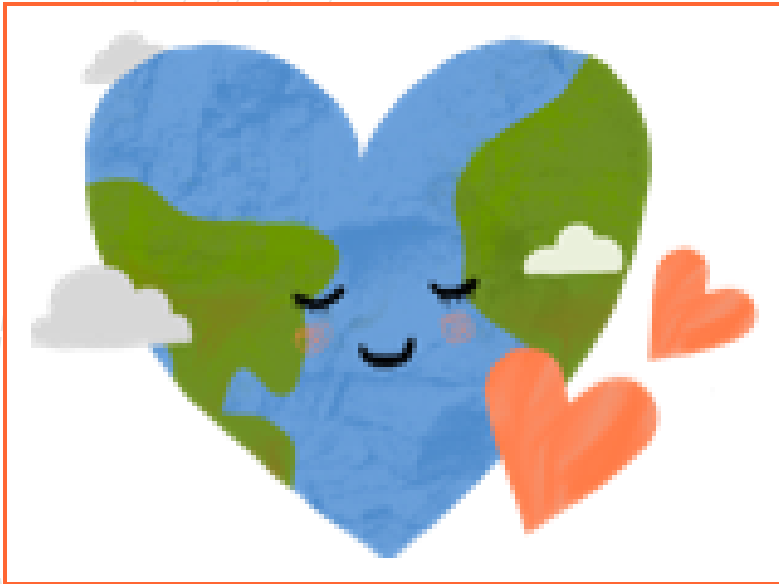
CONSIDERATIONS FOR MODEL DEVELOPMENT AND DEPLOYMENT

How can informed decisions help reduce emissions? (some examples)

- ❖ Running job at a specific time and region to reduce emissions
- ❖ If (Region A = \uparrow Cost & \downarrow Emission
Vs
Region B = \downarrow Cost & \uparrow Emission):
Then Run Job in Region B and use the remaining
amount to reduce emissions.



FUTURE DIRECTIONS



Scopes of emissions.



Improving the carbon transparency of research



Reduction practices
scope-enabled emissions.



Developing certification systems for “Green AI”



Supporting improved estimates of emissions rates.

TBD

TBD

TBD

TBD

Visionary

Energy and Policy Considerations for Modern Deep Learning Research (2020 AAI)

- report training time and sensitivity to hyperparameters
- equitable access to computation resources
- prioritize computationally efficient hardware and algorithms
- **mindful of energy sources powering their compute**

Consumer	Renewable energy consumption
China	22%
Germany	40%
United States	17%
Amazon AWS	50%
Google [†]	100%
Microsoft	50%

Not all cloud services provide equally sustainable compute resource

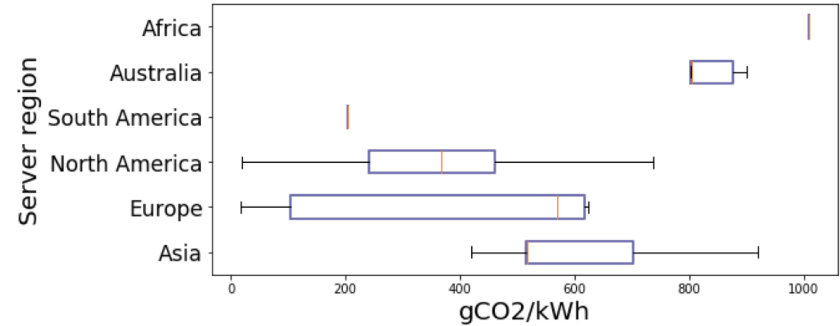
Not possible for all region to source renewable energy

Possible to conduct research about relationship between energy sources and energy usage

Quantifying the Carbon Emissions of Machine Learning (2019 NeurIPS Workshop)

Quantifying the Carbon Emissions of Machine Learning (2019 NeurIPS Workshop)

- illustrate the degree of variability that exists depending on the location of a given server



- Sampled from known GPU server locations from the three major cloud providers: Google Cloud Platform, Microsoft Azure and Amazon Web Services
- Based on pure estimation
- ML Emissions Calculator <https://mlco2.github>

Hardware type: TPUv2 Chip | Hours Used: 100 | Provider: Google Cloud Platf | Region of Compute: us-west1

COMPUTE

CARBON EMITTED	CARBON ALREADY OFFSET BY PROVIDER
6.63	6.63
kg CO ₂ eq. ?	

Power consumption x Time x Carbon Produced Based on the Local Power Grid:

$221\text{W} \times 100\text{h} = 22.1\text{ kWh} \times 0.3\text{ kg eq. CO}_2/\text{kWh} = 6.63\text{ kg eq. CO}_2$

Had this model been run in Google Cloud Platform's **europewest6** region, the carbon emitted would have been of **0.35** kg eq. CO₂

PUBLISH THIS!

Attract More Attention from Researcher

Question: Training BERT-base -> How much CO₂? -> What does this means?

Problem:

- Lack of Visibility
 - Where can I know Carbon footprint?
- What these numbers mean?
 - How to interpret CO₂ weight?

Visible Carbon Footprint

Put to somewhere more noticeable:

- PaperwithCode
- Huggingface
- Github

The screenshot shows the GitHub repository page for 'google-research/multilingual-t5'. The repository is public and has 19 watchers, 103 forks, and 953 stars. It is licensed under Apache-2.0. The repository contains several files: 'multilingual_t5' (refactor tasks), 'CONTRIBUTING.md' (internal change), 'LICENSE' (internal change), and 'README.md' (update mT5 paper link). The README file is expanded to show the following content:

mT5: Multilingual T5

Multilingual T5 (mT5) is a massively multilingual pretrained text-to-text transformer model, trained following a similar recipe as T5. This repo can be used to reproduce the experiments in the [mT5 paper](#).

Table of Contents

- [Languages covered](#)
- [Results](#)
- [Usage](#)
 - [Training](#)
 - [Fine-Tuning](#)
- [Released Model Checkpoints](#)
- [How to Cite](#)

Languages covered

mT5 is pretrained on the [mC4](#) corpus, covering 101 languages:

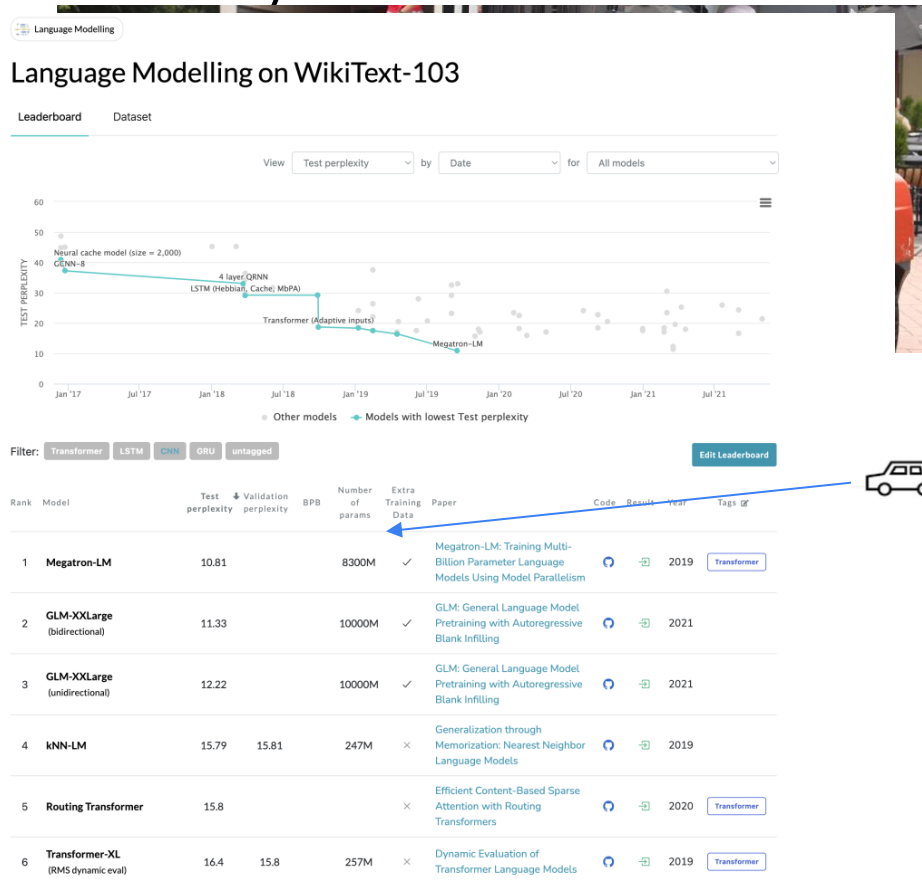
Afrikaans, Albanian, Amharic, Arabic, Armenian, Azerbaijani, Basque, Belarusian, Bengali, Bulgarian, Burmese, Catalan, Cebuano, Chichewa, Chinese, Corsican, Czech, Danish, Dutch, English, Esperanto, Estonian, Filipino, Finnish, French, Galician, Georgian, German, Greek, Gujarati, Haitian Creole, Hausa, Hawaiian, Hebrew, Hindi, Hmong, Hungarian, Icelandic, Igbo, Indonesian, Irish, Italian, Japanese, Javanese, Kannada, Kazakh, Khmer, Korean, Kurdish, Kyrgyz, Lao, Latin, Latvian, Lithuanian, Luxembourgish, Macedonian, Malagasy, Malay, Malayalam,

The right sidebar of the repository page shows the 'About' section with a description, license, and star/fork counts. Below that are sections for 'Releases', 'Packages', 'Contributors', and 'Languages'. The 'Languages' section shows a bar chart with Python at 96.4% and Shell at 3.6%.

Carbon Comparison for ML Community

Find something comparable:
How to interpret Calories in Diet?

Drive car for an hour, etc.



Visionary

- More accurate estimation/quantification technique on carbon emissions for deep learning tasks
- Possible way to estimate the power/carbon emission of TPUs

Carbon Emissions and Large Neural Network Training (<https://arxiv.org/abs/2104.10350>)

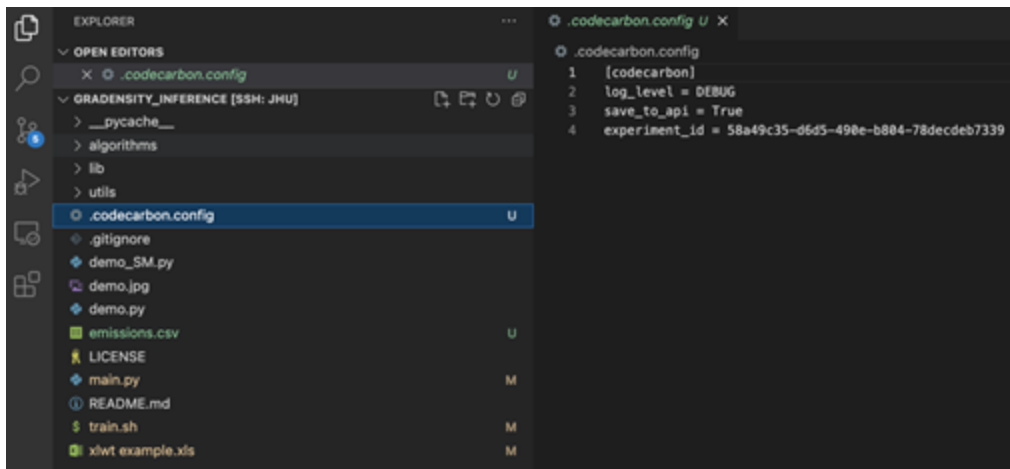
Model	Evolved Transformer NAS	T5	Meena	Gshard -600B	Switch Transformer	GPT-3
Developer	Google					OpenAI
Datacenter of original experiment	Google Georgia	Google Taiwan	Google Georgia	Google North Carolina	Google Georgia	Microsoft
Processor	TPU v2		TPU v3			V100
Energy Consumption (MWh)	7.5	85.7	232	24.1	179	1,287
% of Google 2019 total energy consumption (12.2 TWh = 12,200,000 MWh) [Goo20]	0.00006%	0.00070%	0.00190%	0.00020%	0.00147%	0.01055%
Gross tCO ₂ e for Model Training	3.2	46.7	96.4	4.8	72.2	552.1
Net tCO ₂ e for Model Training	3.2	46.7	96.4	4.3	59.1	552.1

Empiricist

- “Measuring the Carbon Intensity of AI in Cloud Instances” **doesn't publish code yet**
- Try [CodeCarbon](#), a python package framework to calculate emissions
 - Two versions: offline (own server) and online (cloud)
 - $\text{CO}_2 \text{ emissions} = C * P$,
 - where:
 - C = Carbon Intensity of the electricity consumed for computation (kgCO₂/kWh)
 - P = Power Consumed by the computational infrastructure (kWh)
- Experiment: Test CO₂ emission of a python program on a JHU server
 - Input: A python program
 - Output: The CO₂ emissions from the process

Installation and configuration

- Installation:
 - `!conda install -c conda-forge codecarbon`
- Configuration:
 - Create "experiment_id": `!codecarbon init`
 - This id will be used to send results to codecarbon database

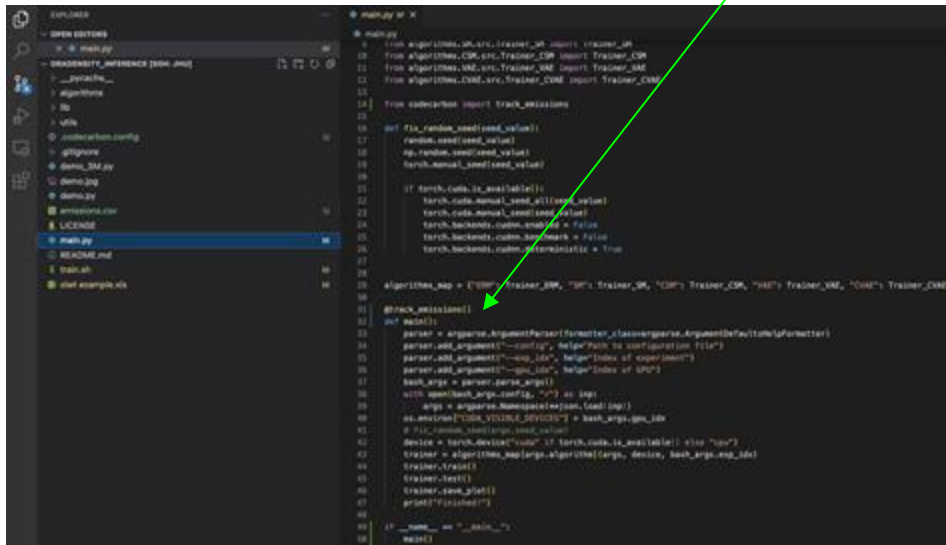


The screenshot shows a code editor interface with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with files like `demo.py`, `main.py`, and `train.sh`. The `.codecarbon.config` file is selected in the explorer and its content is displayed in the editor. The content of the `.codecarbon.config` file is as follows:

```
[codecarbon]
log_level = DEBUG
save_to_api = True
experiment_id = 58a49c35-d6d5-490e-b804-78decdeb7339
```

Installation and configuration

- Configuration:
 - Create "experiment_id": !codecarbon init
 - Track emission from the code by adding "@track_emissions()" before main function, then run the program as usual

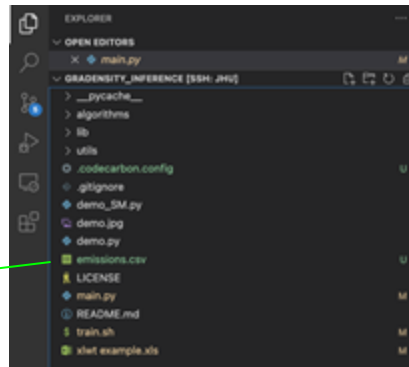


```
1 from algorithms.src.trainer import Trainer_SM
2 from algorithms.src.trainer import Trainer_DM
3 from algorithms.src.trainer import Trainer_CM
4 from algorithms.src.trainer import Trainer_CDM
5
6 from codecarbon import track_emissions
7
8 def fix_random_seed(seed_value):
9     random.seed(seed_value)
10    np.random.seed(seed_value)
11    torch.manual_seed(seed_value)
12
13    if torch.cuda.is_available():
14        torch.cuda.manual_seed_all(seed_value)
15        torch.cuda.manual_seed(seed_value)
16        torch.backends.cudnn.enabled = False
17        torch.backends.cudnn.benchmark = True
18
19 algorithm_map = {"SM": Trainer_SM, "DM": Trainer_DM, "CM": Trainer_CM, "CDM": Trainer_CDM}
20
21 @track_emissions()
22 def main():
23     parser = argparse.ArgumentParser(formatter_class=argparse.ArgumentDefaultsHelpFormatter)
24     parser.add_argument("--config", help="Path to configuration file")
25     parser.add_argument("--exp_dir", help="Index of experiment")
26     parser.add_argument("--exp_id", help="Index of exp")
27     args = parser.parse_args()
28
29     with open(args.config, "r") as f:
30         args = argparse.Namespace.fromkeys(f.readlines())
31         os.environ["CUDA_VISIBLE_DEVICES"] = args.cuda_device_id
32         if fix_random_seed(args.seed_value):
33             device = torch.device("cpu" if torch.cuda.is_available() else "cpu")
34             Trainer = algorithm_map[args.algorithm](args, device, args.exp_id)
35             Trainer.train()
36             Trainer.test()
37             Trainer.save_plot()
38             print("finished")
39
40 if __name__ == "__main__":
41     main()
```

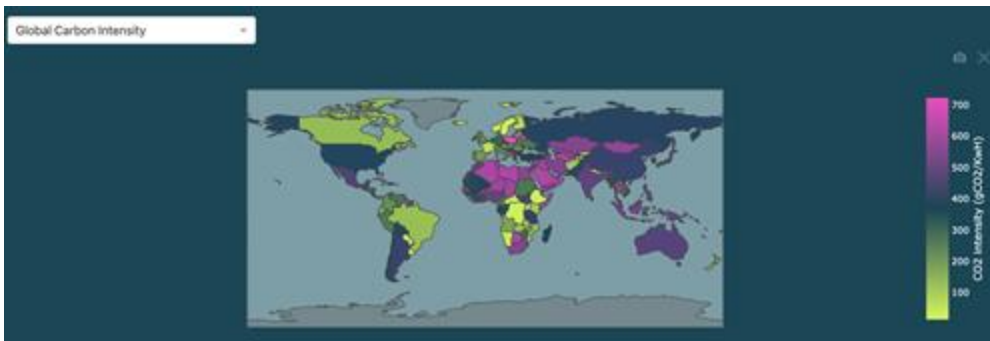
Results

Outputs:

- A csv file, contain tracking CO2 emissions info
- A visualization of CO2 emissions on <https://dashboard.codecarbon.io/>

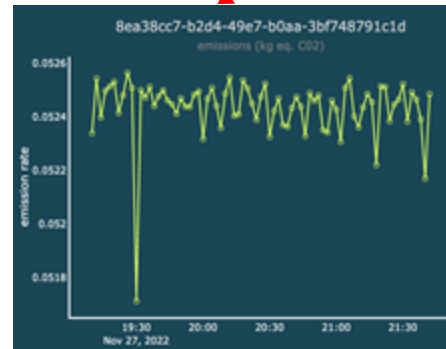
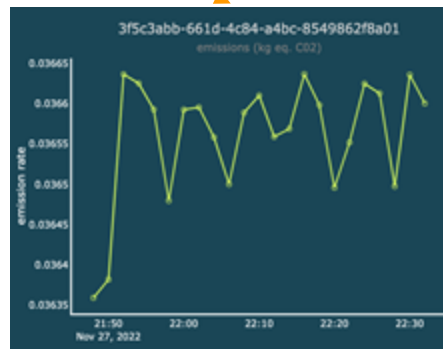
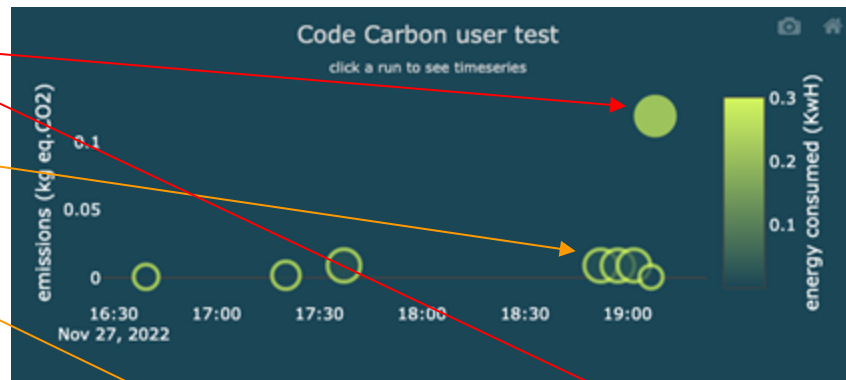


	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	timestamp	project_name	run_id	duration	emissions	emissions_rate	cpu_power	gpu_power	ram_power	cpu_energy	gpu_energy	ram_energy	energy_consumed	country_name
2	2022-11-27 11:45:41	codecarbon	1becddc9-b9ab-478	249.6882091	0.008966197114	0.03590957358	140	91.605	94.33737803	0.009709982558	0.006343394994	0.006538602893	0.02259198045	United States
3	2022-11-27 12:15:44	codecarbon	eb02cb36-5f5-4385	249.4710832	0.008896021137	0.03565952825	140	90.568	94.33737803	0.0097015406	0.006181013685	0.006532604896	0.02241515918	United States
4	2022-11-27 12:21:11	codecarbon	5374d368-1611-427	53.1477232	0.001869655549	0.0351784693	140	86.926	94.33737803	0.002066826603	0.001252569515	0.001391544436	0.004710940554	United States
5	2022-11-27 12:25:52	codecarbon	99b00296-807a-4fb3	249.0747089	0.008938152096	0.03588542624	140	96.047	94.33737803	0.009686132018	0.006312639906	0.006522543947	0.02252131587	United States
6	2022-11-27 12:41:28	codecarbon	155f2b24-53b8-40a2	254.2241695	0.009060454863	0.03563962813	140	90.135	94.33737803	0.009886381733	0.006285629252	0.00665746917	0.02282948016	United States



Results: Comparison between backbones

- Comparison between **Wide ResNet-28-10** and **LeNet5**
- Observation:
 - Wider ResNet-28-10 is **deeper**, therefore run **longer** and cause **much higher CO₂ emission!**



Conclusion

- [CodeCarbon](#), a python package framework to calculate emissions
 - Easy to install and configure
 - Easy to integrate with our code
 - Easy to track CO₂ emissions
- Highly recommend to try if you care about environmental issue!

