



Optimised Deep Learning - Jean Zay

Practical work (of your choice !)



Like you want !

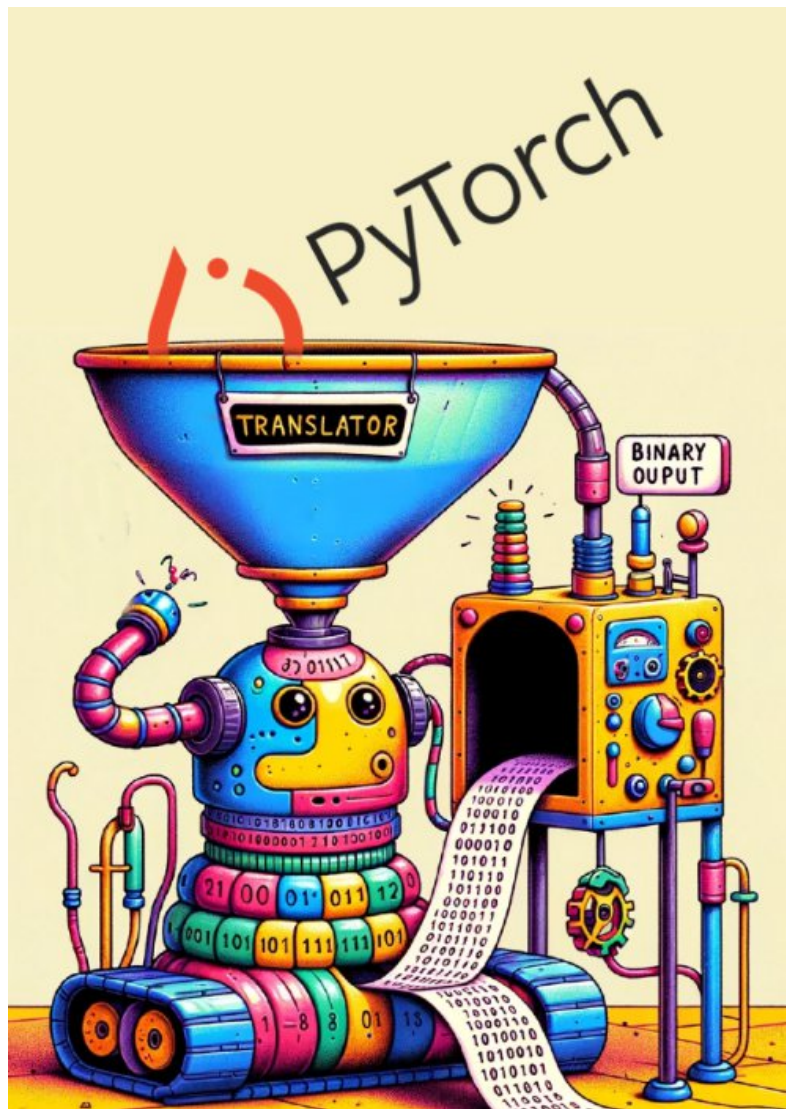
JIT ◀

Data Parallelism under the hood ◀

HPO ◀

Data Augmentation ◀

Large model ◀



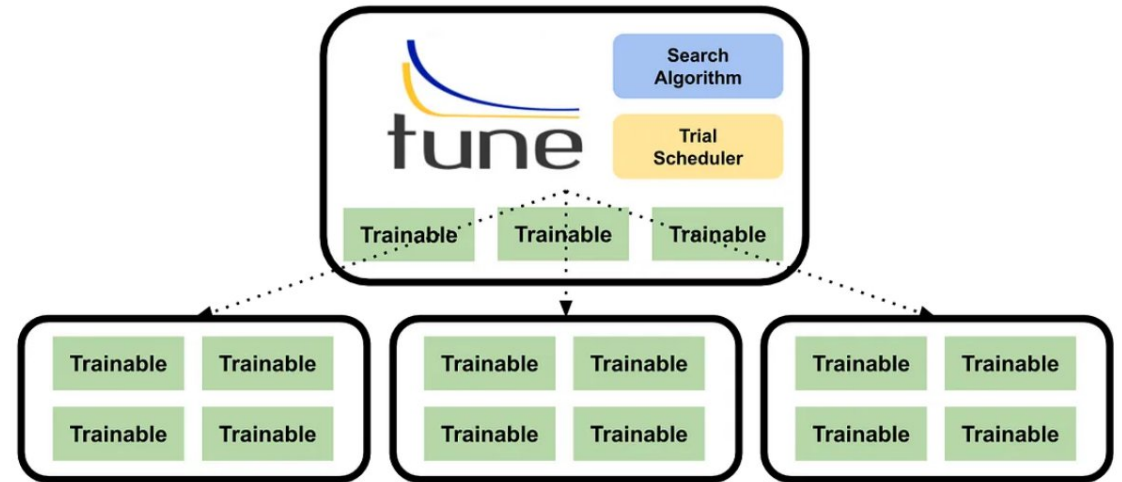
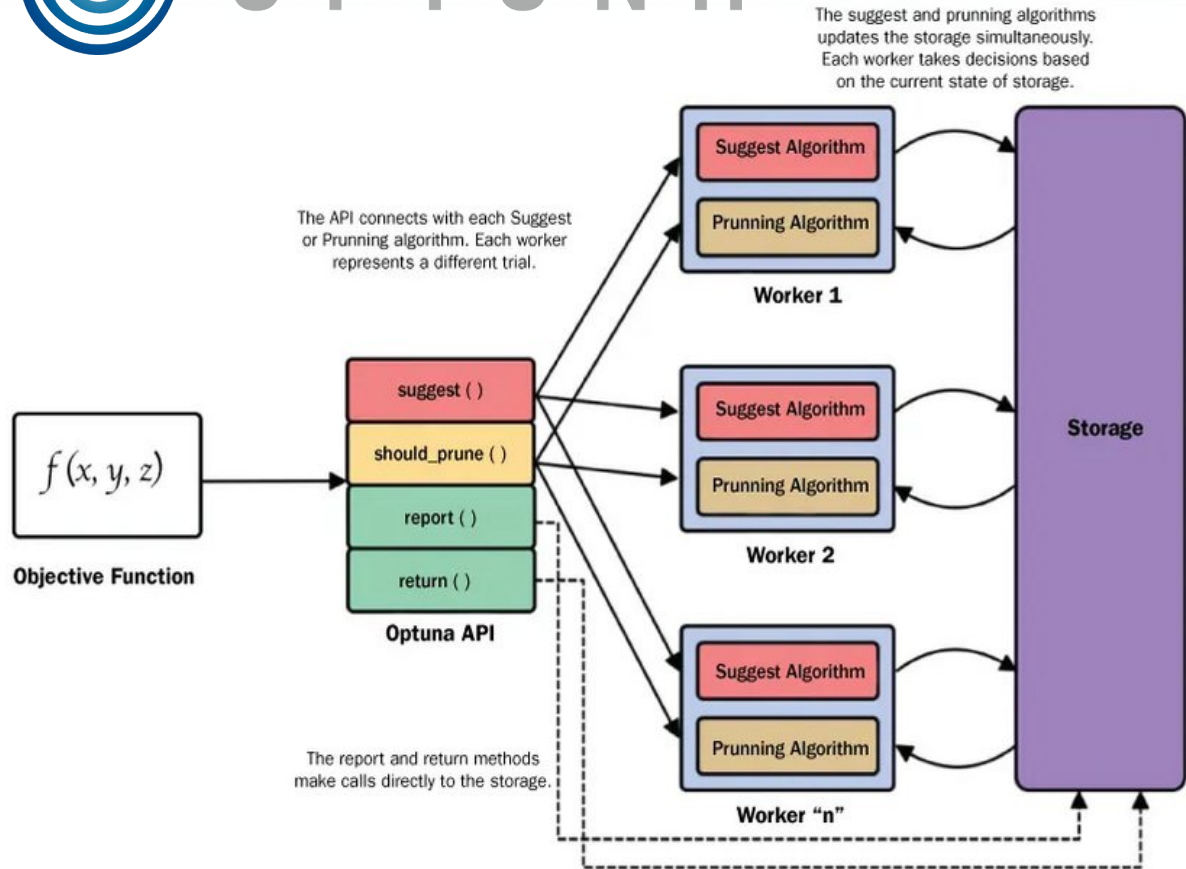
How to accelerate deep learning codes using *old school* compilation ?



How to scale your HPO on supercomputer ? How to use Optuna & Ray at scale ?



OPTUNA



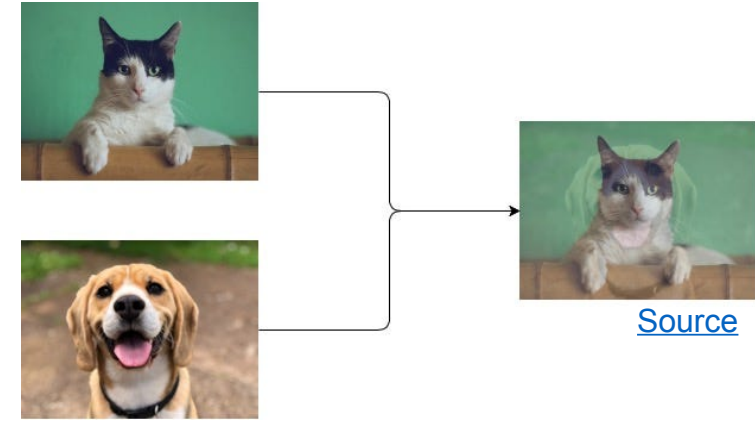
Is CPU enough? How to delegate the Data Augmentation to the GPU? How to optimize the Data Augmentation on the GPU?

- **RandAugment** (torchvision): combinations of multiple transforms, either geometric or photometric, or both



- **MixUp** (*mixup.py*): mixing up the features and their corresponding labels

$$\tilde{x} = \lambda x_i + (1 - \lambda)x_j, \quad \text{where } x_i, x_j \text{ are raw input vectors}$$
$$\tilde{y} = \lambda y_i + (1 - \lambda)y_j, \quad \text{where } y_i, y_j \text{ are one-hot label encodings}$$



- **CutMix** (*cutmix.py*): replacing an image region with a patch from another training image



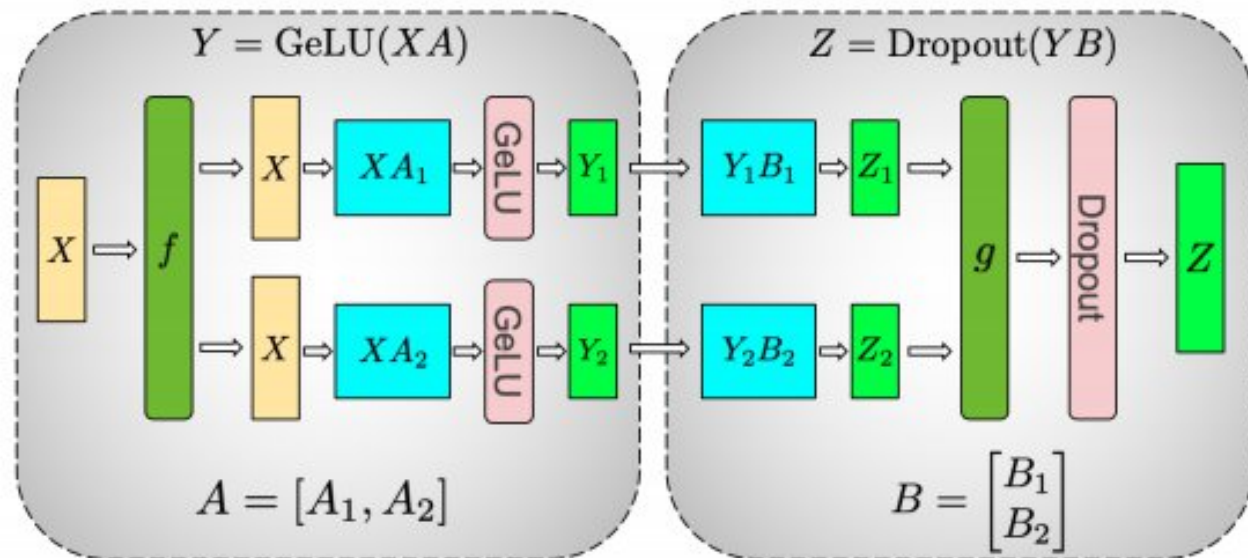
Pipeline Parallelism



Optimized Data Parallelism
ZeRO - FSDP



Tensor Parallelism under the hood



Conclusions

JIT ◀

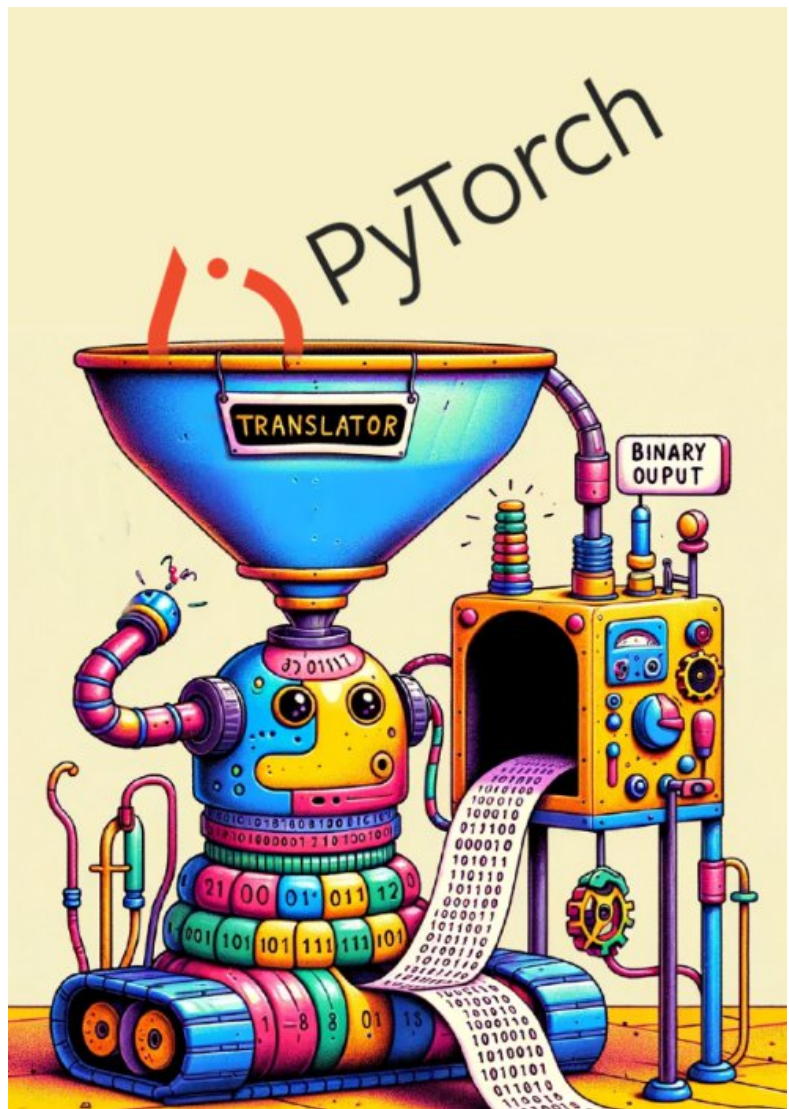
Data Parallelism under the hood ◀

HPO ◀

Data Augmentation ◀

Large model ◀

How to accelerate deep learning codes using *old school* compilation ?



```
model_opt = torch.compile(model, mode="reduce-overhead")
```



How to scale your HPO on supercomputer ? How to use Optuna & Ray at scale ?



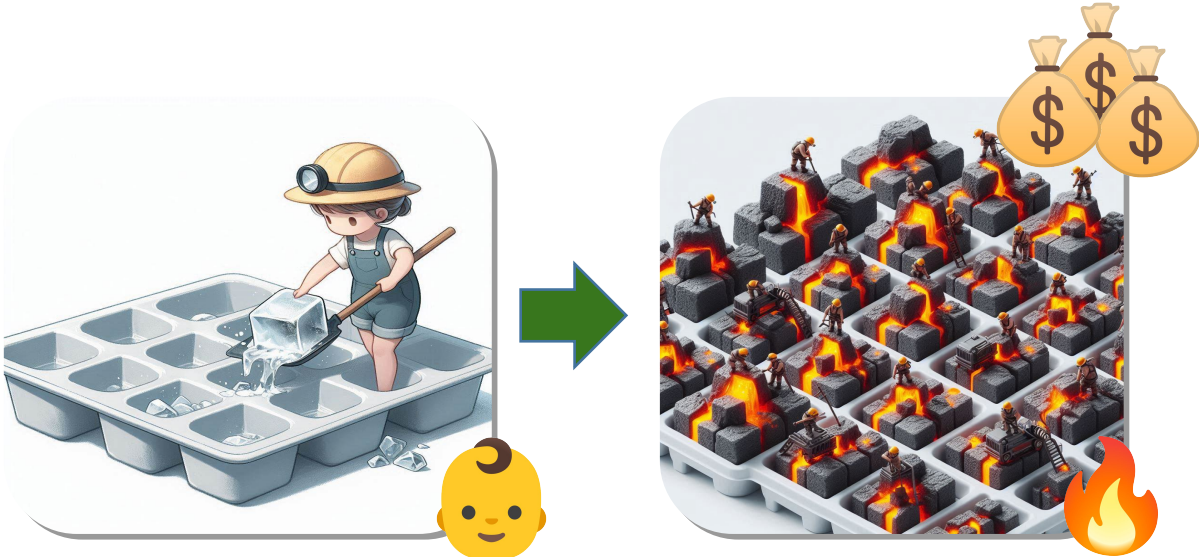
- Database approach
- Not Multiprocess Native



RAY



- Multi-worker native
- Many parallel algorithms
- Difficult to use (especially with slurm)



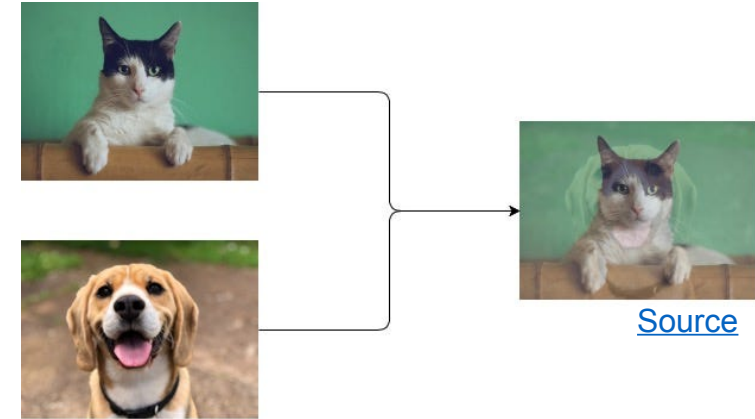
Is CPU enough? How to delegate the Data Augmentation to the GPU? How to optimize the Data Augmentation on the GPU?

- **RandAugment** (torchvision): **CPU is enough**



- **MixUp** (*mixup.py*): **much better on GPU**

$$\tilde{x} = \lambda x_i + (1 - \lambda)x_j, \quad \text{where } x_i, x_j \text{ are raw input vectors}$$
$$\tilde{y} = \lambda y_i + (1 - \lambda)y_j, \quad \text{where } y_i, y_j \text{ are one-hot label encodings}$$



- **CutMix** (*cutmix.py*): **transform loop over images into loop over batches to improve parallelism and benefit from the GPU acceleration**



Large model

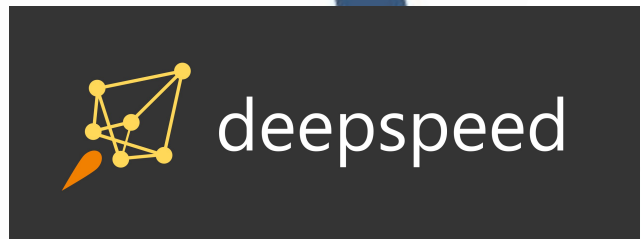


Pipeline Parallelism
from scratch

Pipeline Parallelism
Deepspeed
Implementation

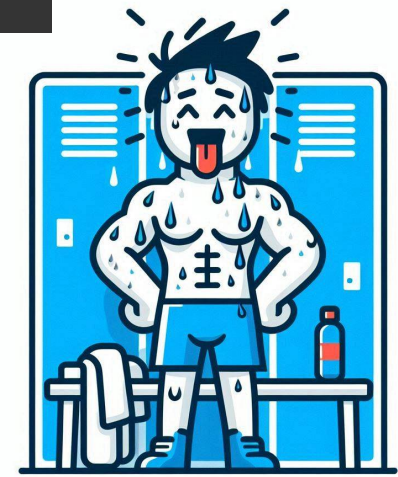


PyTorch

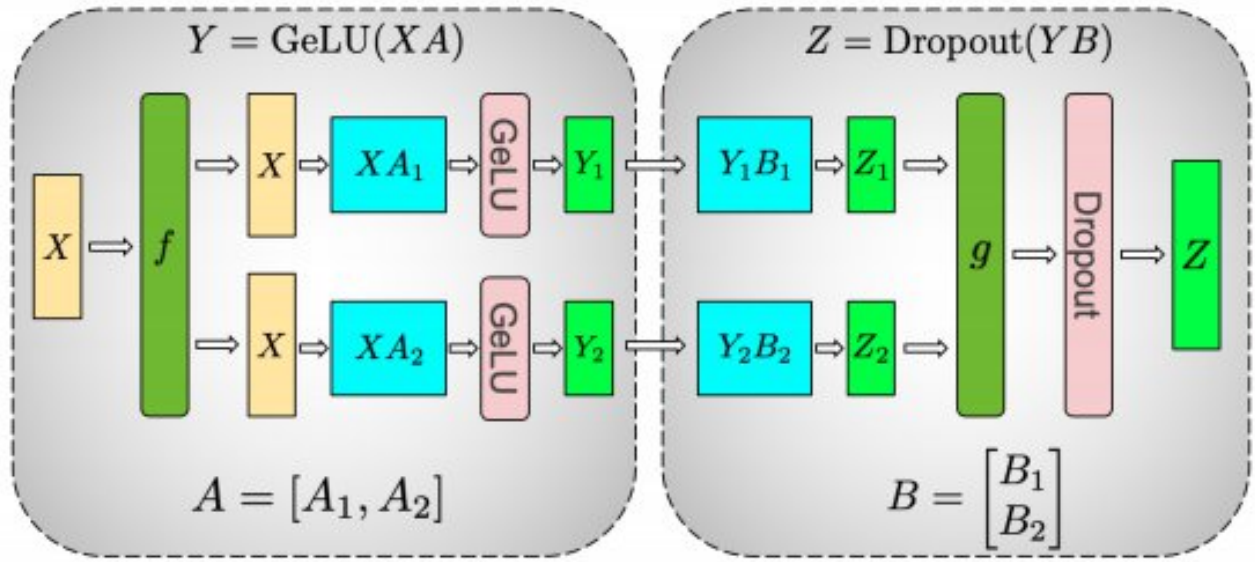



FSDP












ZeRO



Tensor Parallelism under the hood



 **tp tensor slicing**
 Nathan CASSEREAU authored 3 months ago

Name	Last commit
..	
 balise1.py	tp tensor slicing
 balise10.py	tp tensor slicing
 balise11.py	tp tensor slicing
 balise2.py	tp tensor slicing
 balise3.py	tp tensor slicing
 balise4.py	tp tensor slicing
 balise5.py	tp tensor slicing
 balise6.py	tp tensor slicing
 balise7.py	tp tensor slicing
 balise8.py	tp tensor slicing
 balise9.py	tp tensor slicing