Explore the Possibility of Fine-grained Non-encrypted Distributed MLaaS: an Adversary View

Final Talk for *CyberSec & Privacy*, Type: Research

Bohan Cui

Nanjing University, Department of Computer Science and Technology

Nov 2024

A quick review on the Background: distributed MLaaS

- ML models requires more and more devices and power. Therefore MLaaS is a trend.
- (In another proj) We are going to design a distributed system to utilize the edge devices.
- We try to check (and solve) the IP privacy concern of **this kind of system**.[1]

Fig. 1: WSJ's report on GPT-5's failure

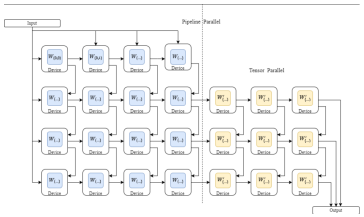
[1] which kind of system? systems having the assumption in the next slide.

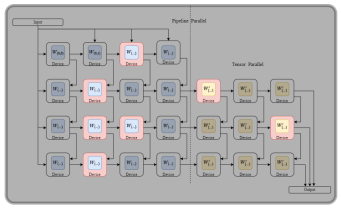
A quick review on the roblem

- Our assumption: The model is uncrypted and the pirates **can not** control all the devices. All the devices are untrusted.
- Our goal: To protect the IP of the model, including getting a copy of the **whole** model or get **similar** performance of the model with efforts **lower than** retraining the model.
- Our task: If protection needed, method. If not, evidence.

Quick review on our logic chain

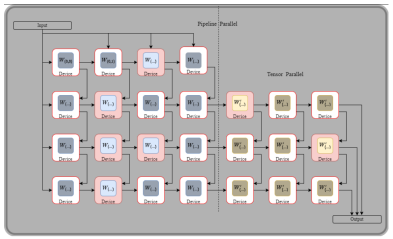
S0:
Basic model for this problem





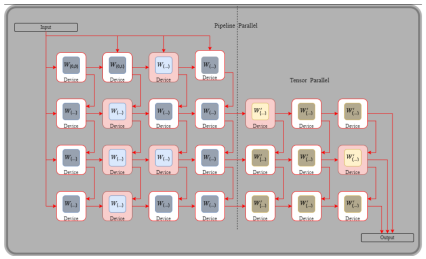
P1:
Just acquire part of the params

S1:
Complexity of the Operators



P2:
heuristic search for places of params

S2:
heuristic search for places of params



S3:
Stream Obfuscation

P3: Stream Hijack

P4: .....

Fig. 2: Logic Chain

Design of the experiment(Review)

- Experiments are conducted on a VGG16 pretrained on CIFAR-100 (baseline acc: about 61%).
- We manipulate the largest MLP layer of the model, a 25088×4096 dense layer, and we cut it into 16×98 blocks (256×256 each).

Design of the experiment(Review)

- On P_1: We randomly prune part of the blocks and test the accuracy of the model to show the pirate gets parts of the params with full knowledge of their position.

- On S_1 We randomly exchange the position of the reserved blocks and test the accuracy of the model to show the pirate gets parts of the params without knowledge of their position in a certain layer.

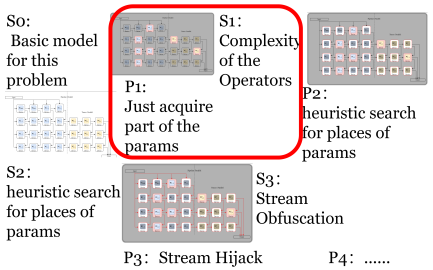


Fig. 3: Exp1 and Exp2

Design of the experiment(New)

- On P_2:
- Abstraction: Given a function $\mathbf{Val}$, a set of values S, among all the orders of S, find $max(\mathbf{Val}(\mathbf{order}(\mathbf{S})))$
- It is an combinatorial optimization problem. We use some heuristic algorithms to solve it.
- Genetic Algorithm: Any position combination of the blocks is an object in the population. And the fitness function is the accuracy of the model. Objects evolve by exchange the position of the blocks.
- Reinforcement Learning: The agent is the pirate. The environment is the model evaluation. The action is the exchange of the position of the blocks. The reward is the accuracy of the model.

Not yet: On P_3

- I tried to construct a pattern recognition model to solve it. I found it is hard:[3]
- It is hard to conduct recognition on the topology of the devices. (A upper triangle matrix, what is neighbor?)
- Different design have totally different stream patterns. It is hard to classify them.
- If time permits, I plan to turn to traditional algorithm on some property of the topo graph, which is more feasible.

[3] This part has been less important due to the result of P_2.

Review on P_1 and S_1

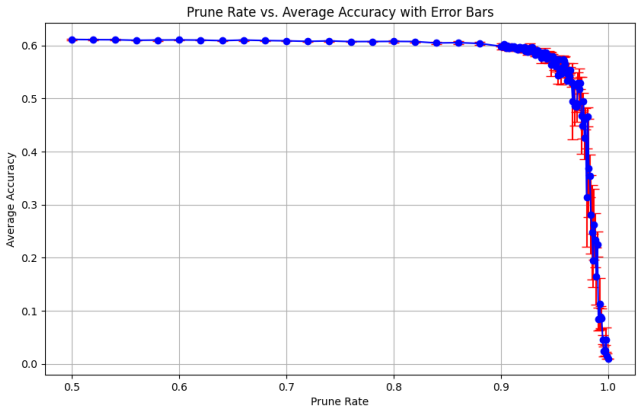


Fig. 4: Exp1[a]

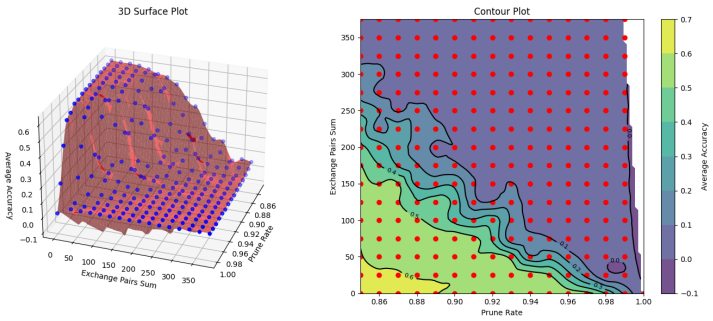


Fig. 5: Exp2

[a] It is a strange result here (after teacher's review). We will work on it. But we may assume there is always a threshold to be 'relatively good' in this pre.

New Results on P_2

- Can pirates get a acceptable accuracy by heuristic searching in the combinations?
- Sorry, NO. I tested the population from 10 to 100 to try to find a good combination.[4] However, it was always 1% percent accuracy just from the first generation.(1% in CIFAR-100 absolutely means nothing.)
- When population is 100, it should run 10000 times evaluation to finish the algorithm, with the evaluation set with 100 pics. It needs 5000 GPUminutes on RTX-4060, which is 27x of the training of the model on RTX-4060 (2.5 GPUhours)! [5]

[4] actually larger than 100 is absolutely better, but it is too slow.

[5] and actually more scale is not so meaningful that we can normally train a even better model with 10000 steps with batchsize being 100!

Some insight into this result (1)

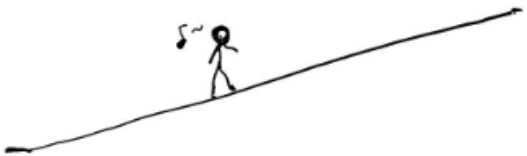
- We can give some explanations to this result. Why NOT?
- The space is too large:
  Go(AlphaGo): $361!$, Our problem: $1568!$
- And the initial state of the algorithm needs some enough diversity.[6]
- What about the possibilty of catch any possible clue?
- We assume the least standard that if any block is in its original position and control the feature with no inference, it could make a difference to the result.[7]
  $P(clue) = \frac{1}{16 \times 98} \times (\frac{15}{16})^{(156-1)}$
  $E(1\ clue\ in\ all\ population) = 0.000452$

[6] or may be called clues, implicit patterns ... if you like

[7] and you can find that if you count on variation to generate clues, the P will be even much much smaller.

Some insight into this result(2)

- Much Worse:
- The discrete envaluation. 'A bit correctness' is far than a variation in the accuracy on the envaluation set.(100 pics for 100 types).
- So in the generation, all the accuracy is 1% even if some of them is 'a bit correct'.[8]
- Optimization: You can never optimize on a general 'plate'. To make it not so even, you need more batch size. [9]



[8] You may think you can compare all the features in the vector and get a loss, but no, it will be more sensitive and make it never converge.

[9] Here is a balance.

Conclusion

- Given the pirate cannot clone the model even if they know the architecture of the model (i.e. which layer do the parameters belong to), We can give a conclusion that we can deploy the model in **this kind of system** without encryption.
- The foundation of the conclusion is the fine-grained. We must split the parameters into fine-grained blocks which will sacrifice the performance of the system.[10] Therefore, we need to evaluate the possibility in a certain real system.

[10] It may have a threshold in 'how fine', and I can work on it.

the future of the problem: may not exist[12]

- Recent news and Ilya's argument shows that the pre-trained model is going to the dead end due to the limited data.
- Now: The training task cannot be done by a small company.
- Future: The training task cannot be done by any single company.
- A direct conclusion is the most powerful models will be open source and IP is not so important.[11]
- And the relatively small models are getting smaller and smaller, which enable them to run on single devices. (*Densing Law of LLMs, Chaojun Xiao et.al.*)

[11] Otherwise, they will be Cartel and monopolize the market.

[12] but definitely exist in this project

Thanks!