AutoDA: Automated Decision-based Iterative Adversarial Attacks

Qi-An Fu, *Dept. of Comp. Sci. and Tech., Institute for AI, Tsinghua-Bosch Joint ML Center, THBI Lab, BNRist Center, Tsinghua University, Beijing, China;* Yinpeng Dong, *Dept. of Comp. Sci. and Tech., Institute for AI, Tsinghua-Bosch Joint ML Center, THBI Lab, BNRist Center, Tsinghua University, Beijing, China; RealAI;* Hang Su, *Dept. of Comp. Sci. and Tech., Institute for AI, Tsinghua-Bosch Joint ML Center, THBI Lab, BNRist Center, Tsinghua University, Beijing, China; Peng Cheng Laboratory; Tsinghua University-China Mobile Communications Group Co., Ltd. Joint Institute;* Jun Zhu, *Dept. of Comp. Sci. and Tech., Institute for AI, Tsinghua-Bosch Joint ML Center, THBI Lab, BNRist Center, Tsinghua University, Beijing, China; RealAI; Peng Cheng Laboratory; Tsinghua University-China Mobile Communications Group Co., Ltd. Joint Institute;* Chao Zhang, *Institute for Network Science and Cyberspace / BNRist, Tsinghua University*

https://www.usenix.org/conference/usenixsecurity22/presentation/fu-qi

This artifact appendix is included in the Artifact Appendices to the Proceedings of the 31st USENIX Security Symposium and appends to the paper of the same name that appears in the Proceedings of the 31st USENIX Security Symposium.

August 10–12, 2022 • Boston, MA, USA

978-1-939133-31-1

Open access to the Artifact Appendices to the Proceedings of the 31st USENIX Security Symposium is sponsored by USENIX.

A Artifact Appendix

A.1 Abstract

The most important experiment for our work is the searching for programs experiments described in Section 5.1. This experiment involves a binary compiled from C++ source code called `autoda`. It requires about 20 CPU cores and one GPU with at least 8 GiB of GPU memory to run efficiently. The ablation study experiments described in Section 5.4 is also a binary compiled from C++ source code, and have similar requirement with the `autoda` binary. The benchmark experiments requires GPUs. Its entry point is a python script.

The whole experiments in Section 5.1 and Section 5.4 should spend about one hundred GPU hours. The whole benchmark experiments in Section 5.2 and Section 5.3 should spend about several hundreds GPU hours. The searching for programs experiments can hardly be exactly reproduced, since randomly generating exactly the same program is extreme rare. However, the lowest ℓ_2 distortion ratio in each run can be reproduced. What's more, we provide the log files where we found the AutoDA 1st and 2nd in the `log/` directory.

A.2 Artifact check-list (meta-information)

- **Program:** See the `README.md` file.
- **Compilation:** See the `README.md` file.
- **Binary:** The `autoda` binary is for Experiment 5.1, and the `autoda_ablation` binary is for Experiment 5.4.
- **Model:** Most of them are pre-trained models provided by previous work, see `README.md`. The DenseNet, DPN, and DLA models are trained by ourselves using public available code at https://github.com/kuangliu/pytorch-cifar.
- **Data set:** The CIFAR-10 dataset and ImageNet dataset. The ImageNet dataset requires pre-processing, see `prepare_models/README.md`.
- **Run-time environment:** GNU/Linux. Need root access to install necessary dependencies.
- **Hardware:** Need GPU with at least 8 GiB of GPU memory. Need CPU with at least 20 cores.
- **Execution:** In our searching experiments described in Section 5.1, we run the same binary `autoda` for 50 times, and each run spends about two hours with one GTX 1080 Ti GPU and about 20 CPU threads. The full benchmark experiments in Section 5.2 and Section 5.3 should spend about several hundreds GPU hours.
- **Metrics and Output:** For the `autoda` binary, it would write the program we found into the log file passed via command line, as well as evaluation metrics (lines started with `rs=`, you could get evaluation metric for each programs by parsing the floating-point number in these lines). Since our searching for programs experiments described in Section 5.1 cannot be exactly reproduced, we provide the two runs' log files in the `log/` directory where the AutoDA 1st and 2nd algorithms are found. For the `autoda_ablation` binary, it would write the ratios into the log file passed via command line (lines started with `ratios_mean=`).
- **Experiments:** Besides the searching for programs experiments and the ablation study experiments, we also have benchmark experiments (Section 5.2 and Section 5.3). These benchmark experiments all use the `prepare_models/attacker.py` script. See `README.md` for more details on the usage of this script. This script would output adversarial example's distance to the original image at each step for each sample into a hdf5 file, thus further analysis could be done. This script would also print out the attack success rate and average/median ℓ_2 distortion during the attack process for the current batch.
- **How much disk space required (approximately)?:** 20 GiB.
- **How much time is needed to prepare workflow (approximately)?:** Setting up environment for compiling the `autoda` binary and the `autoda_ablation` is relative easy, several hours should be enough. Setting up environment for benchmark experiments from scratch are much more complicated, including training some models from scratch, downloading and pre-processing ImageNet dataset, and installing dependencies for every model.
- **How much time is needed to complete experiments (approximately)?:** About one hundreds GPU hours for the search for programs experiments and the ablation study experiments. About several hundreds GPU hours for benchmark experiments.
- **Publicly available (explicitly provide evolving version reference)?:** https://github.com/Fugoes/AutoDA/commit/257cf85e1c0c1d129a50a274764ed6bc893ccde5.
- **Code licenses (if publicly available)?:** MIT.

A.3 Description

A.3.1 How to access

https://github.com/Fugoes/AutoDA/commit/257cf85e1c0c1d129a50a274764ed6bc893ccde5.

A.3.2 Hardware dependencies

Need Nvidia GPU with at least 8 GiB of GPU memory. Need CPU with at least 20 cores.

A.3.3 Software dependencies

See `README.md`.

A.3.4 Data sets

CIFAR-10 and ImageNet.

A.3.5 Models

See `README.md`.

A.3.6 Security, privacy, and ethical concerns

This work does not have security, privacy, or ethical concerns.

A.4 Installation

See `README.md`.

A.5 Experiment workflow

For experiments in Section 5.1 and Section 5.4, first compile the source code:

```
cd␣~/
git␣clone␣git@github.com:Fugoes/AutoDA.git
cd␣AutoDA/
mkdir␣build/
cd␣build/
cmake␣-DCMAKE_BUILD_TYPE=Release␣~/AutoDA
make␣-j`nproc`
```

These commands will build the `autoda` binary and `autoda_ablation` binary.

To run the `autoda` binary,

```
CUDA_VISIBLE_DEVICES=0␣./autoda␣\
␣␣␣␣--dir␣~/path/to/data␣\
␣␣␣␣--threads␣16␣\
␣␣␣␣--gen-threads␣20␣\
␣␣␣␣--class-0␣0␣--class-1␣1␣\
␣␣␣␣--cpu-batch-size␣150␣\
␣␣␣␣--gpu-batch-size␣1500␣\
␣␣␣␣--max-queries␣500000000␣\
␣␣␣␣--output␣5ww_queries_00.log
```

The evaluation metrics would be written to `/path/to/data/5ww_queries_00.log`. To quickly check the lowest ℓ_2 distortion ratios,

```
cat␣5ww_queries_00.log␣|
␣␣␣␣grep␣rs=␣|
␣␣␣␣sort␣|␣head
```

Running the `autoda_ablation` binary is similar to running `autoda`. To quickly check for the top 200 lowest ratios,

```
cat␣ablation.log␣|
␣␣␣␣grep␣'^ratios_mean='␣|
␣␣␣␣awk␣-F␣'='␣"{␣print␣\$2␣}"␣|
␣␣␣␣sort␣-n␣|␣head␣-200
```

As for the benchmark experiments, please check the `README.md` file.

A.6 Evaluation and expected results

Though all attacks have randomness, when running the benchmark experiments, the results should be quite close to our reported results given large enough test set.

A.7 Experiment customization

A.8 Notes

The `autoda` and `autoda_ablation` does not exit cleanly, they would crash themselves when reaching max queries and core dumped.

A.9 Version

Based on the LaTeX template for Artifact Evaluation V20220119.