USENIX'23 Artifact Appendix: Squint Hard Enough: Attacking Perceptual Hashing with Machine Learning

Jonathan Prokos[1*], Neil Fendley[2], Matthew Green[1], Roei Schuster[3], Eran Tromer[4], Tushar M. Jois[1], and Yinzhi Cao[1]

[1]Johns Hopkins University, jprokos4@gmail.com, {jois, mgreen, yzcao}@cs.jhu.edu
[2]Johns Hopkins University Applied Physics Laboratory, fendley@jhu.edu
[3]Vector Institute, roei@vectorinstitute.ai
[4]Tel Aviv University and Columbia University, et2555@columbia.edu

A Artifact Appendix

A.1 Abstract

To the reviewers we provide our entire codebase for running our attacks described from the paper. This is a private GitHub repository which we have packaged into a tarball with a README.md with more information into configuration. This codebase is not released to the public due to ethical reason discussed in A.2.1.

A.2 Description & Requirements

The attack has been tested on a 64-bit Windows machine and on a 2017 Intel Macbook Pro. This limitation is due to the PhotoDNA binary which is architecture specific. Our attacks on PDQ have been tested to work on Linux as well.

To reproduce our major results, you can run our attack as described in the README.md with the appropriate parameters or left blank for default parameters.

A.2.1 Security, privacy, and ethical concerns

One ethical concern is with the protection of the PhotoDNAx64.dll and PhotoDNAx64.so files since these are not publicly released. Additionally, our attack could be modified to use on real world systems and therefore should not be publicly available at the risk of aiding in the actions of a malicious actor[1].

A.2.2 How to access

While we do not provide public access to our codebase as discussed in A.2.1, we do provide perceptualhashing.lol which contains more details regarding our codebase.

[*]Currently affiliated with Two Six Technologies, LLC (Arlington, VA).
[1]Discussed in more detail in §1 of our paper.

A.2.3 Hardware dependencies

The only dependency is to have a machine compatible with the corresponding .dll or .so file for PhotoDNA. This should work on a 64-bit Windows or Intel Mac environment. The rest of our attack (including that on PDQ) works with or without gpu[2].

A.2.4 Software dependencies

The main dependency is pytorch. Additional dependencies listed in requirements.txt.

A.2.5 Benchmarks

Our attack requires the ImageNet 2012 Validation dataset. Instructions to obtain are detailed in the README.md file.

A.3 Set-up

You may install all requirements utilizing our provided setup.py file, simply run *pip install -e .* to do so. For GPU support see the provided README.md for more instruction. Additionally, the ILSVRC2012 Validation Images Dataset tarball and two development kits must be placed in *src/data/imagenet*.

A.3.1 Installation

To install any dependencies run *pip install -e src/*. Then to test functionality you may run *python src/hashattack/hashing/pyphotodna.py* and *python src/hashattack/hashing/pdq_hash_test.py*.

[2]Note that in our evaluation of the PDQ algorithm, we utilize a 72-core machine to achieve our 3-hour runtime. See §5.1 in our paper for more info.

A.3.2 Basic Test

The above mentioned files *pyphotodna.py* & *pdq_hash_test.py* will test the ability to produce a hash with either algorithm. These tests will output a 2-d tensor of integers. To perform a more thorough test with our system you may run *python src/hashattack/hash_atk.py --no-write -cm 10* and *python src/hashattack/fuzzy_collisions_avoidance.py --no-write* which you can kill once it begins looping.

A.4 Evaluation workflow

A.4.1 Major Claims

(C1): We are able to achieve a successful attack on both PhotoDNA and PDQ utilizing our threat model described in §3. This includes producing a Second-Preimage and Collision Avoidance image at the calculated baseline threshold[3].

A.4.2 Experiments

(E1): *Targeted-Second-Preimage Attack [10 human-minutes + 4 compute-hours per run[4] + 20GB disk]:*
How to: To run the experiment run *python src/hashattack/hash_atk.py* using both the *-ha pdna* and *-ha pdq* flag.
Preparation: Install the pip package using *pip install -e src/* and download the ImageNet dataset as described above.
Execution: Run the provided code using the appropriate flags as mentioned in the results section 5.3. Once all 20 images have converged using both algorithms, you may view the results from the produced tensorboard.
Results: The results will be displayed using tensorboard.

(E2): *Detection Avoidance Attack [10 human-minutes + 1 compute-hour]:*
How to: To run the experiment run *python src/hashattack/fuzzy_collisions_avoidance.py* using both the *-ha pdna* and *-ha pdq* flag.
Preparation: Install the pip package using *pip install -e src/* and download the ImageNet dataset as described above.
Execution: Run the provided code using default parameters. This will output the attack progression as an image over three set distance values.
Results: The results can be viewed from the terminal. The final output image will be saved to disk.

[3]Code used to calculate this baseline provided as well.

[4]Attacks on PDQ require significantly longer to run and should be carried out on a high core cpu machine. Discussed further in §5.1 & §5.3.1.

A.5 Notes on Reusability

This artifact may be used for additional study on a wide range of perceptual hash functions, but providing this code publicly could facilitate malicious activity.

A.6 Version

Based on the LaTeX template for Artifact Evaluation V20220926. Submission, reviewing and badging methodology followed for the evaluation of this artifact can be found at `https://secartifacts.github.io/usenixsec2023/`.