Атаки на метрики

Георгий Бычков, Михаил Дремин

Video Group
CS MSU Graphics&Media Lab

FGSM

Градиент

Пусть дана дифференцируемая скалярная функция $f(x_1, \ldots, x_n)$.

Тогда ее градиентом называется вектор

$$\nabla f = \left(\frac{\partial f}{\partial x_1}(x_1, \ldots, x_n), \ldots, \frac{\partial f}{\partial x_n}(x_1, \ldots, x_n)\right)$$

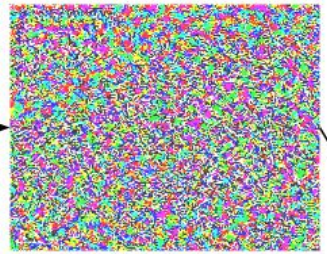
FGSM
Общий вид

x

Атакуемая метрика качества изображений/видео

θ

backpropagation



$sign(\nabla_x \theta(x))$

$x_{adv} = x + \epsilon \times sign(\nabla_x \theta(x))$

FGSM

Оптимизируемая функция

Классификаторы **vs** **Метрики**

$$loss = -CrossEntropyLoss(target\_class, \theta(x))$$

$$loss = 1 - \frac{\theta(x)}{range(\theta)}$$

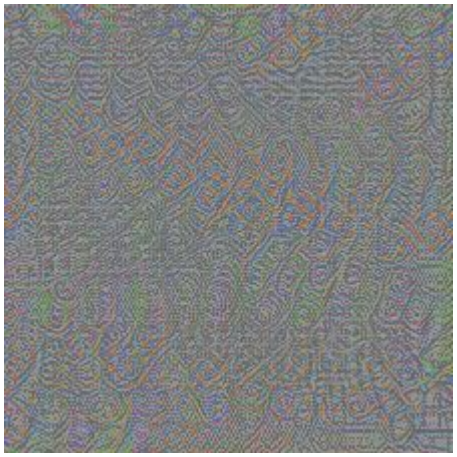
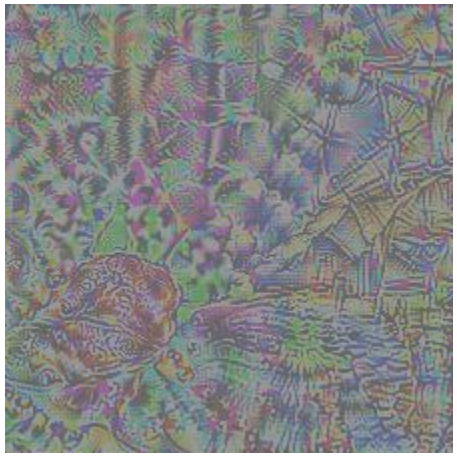
Для метрик: $$\nabla loss = -\frac{\nabla \theta(x)}{range(\theta)}$$

UAP атака

Реализация в простейшем случае

- Необходима обучающая выборка из достаточно большого количества разнообразных изображений.
- Пусть *p* - обучаемая универсальная надбавка, а *batch* - некоторое количество изображений из выборки.

$$loss = 1 - \frac{\theta(batch + p)}{range(\theta)} \qquad p_{new} = p_{old} - \alpha \nabla loss$$


Дифференцируемый JPEG

Напоминание JPEG

Дифференцируемый JPEG
Новшества

Теперь JPEG дифференцируемый!

То есть его можно встроить в наши атаки!

Дифференцируемый JPEG

Как пользоваться

```python
from DiffJPEG import DiffJPEG
jpeg = DiffJPEG(height=224, width=224, differentiable=True, quality=80)
compressed_diff_jpeg = jpeg(im_tensor)
```

✓ 0.0s Python

Задание по атакам на метрики

Описание

- Дан обучающий датасет из изображений.
- Метрика качества изображений - PaQ-2-PiQ.
- Необходимо сделать 2 атаки: итеративную (например, FGSM) и UAP.
- Проатакованные изображения должны быть устойчивы к сжатию.

Задание по атакам на метрики

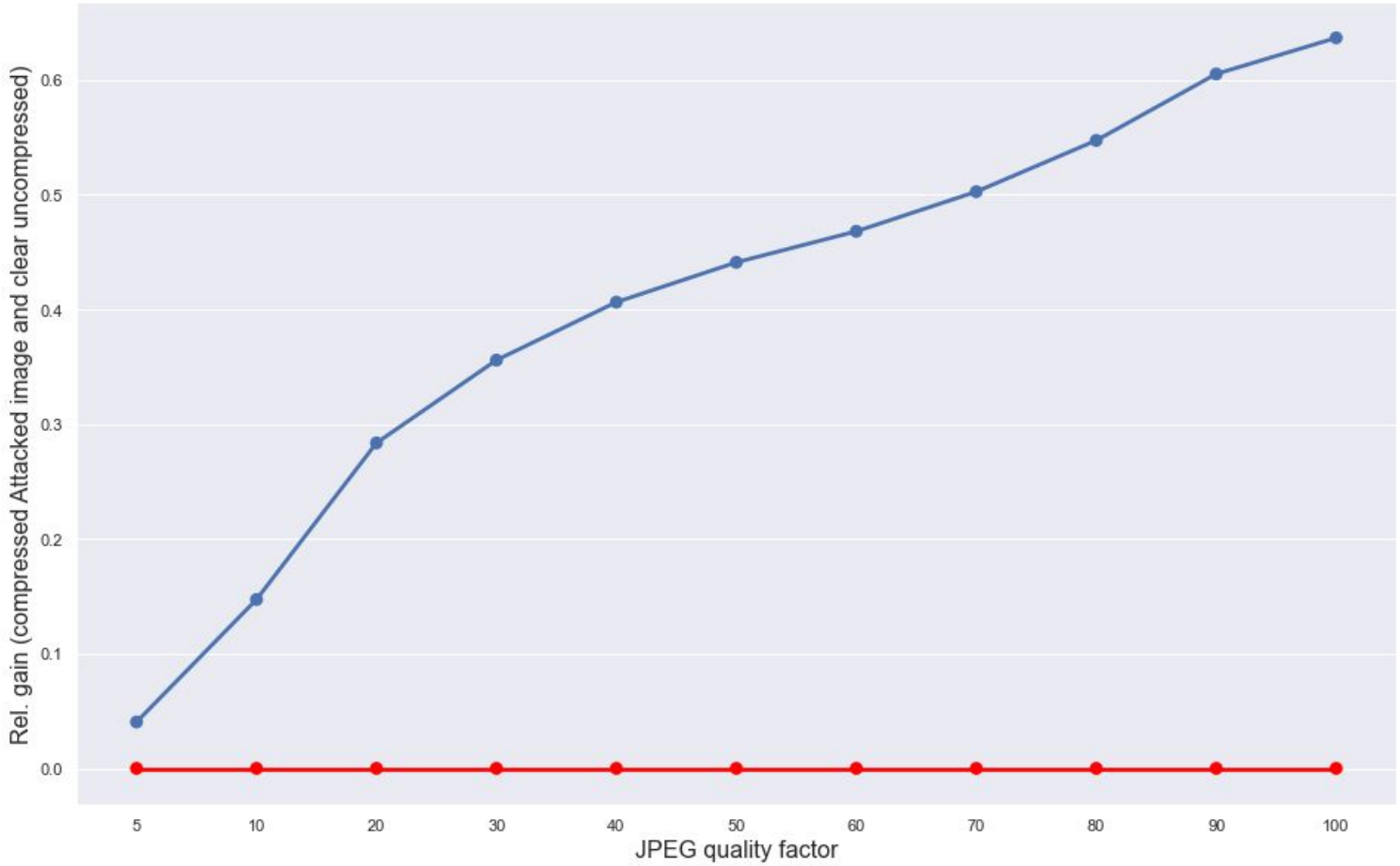
Оценивание (1)

- Для каждого изображения вычисляется площадь под графиком с абсциссой JPEG quality factor и ординатой Rel. gain (отношение значения PaQ-2-PiQ на проатакованном изображении к чистому, минус 1 для центрирования) для каждого

$$\varepsilon \in \{\frac{2}{255}, \frac{4}{255}, \frac{8}{255}, \frac{10}{255}\}.$$

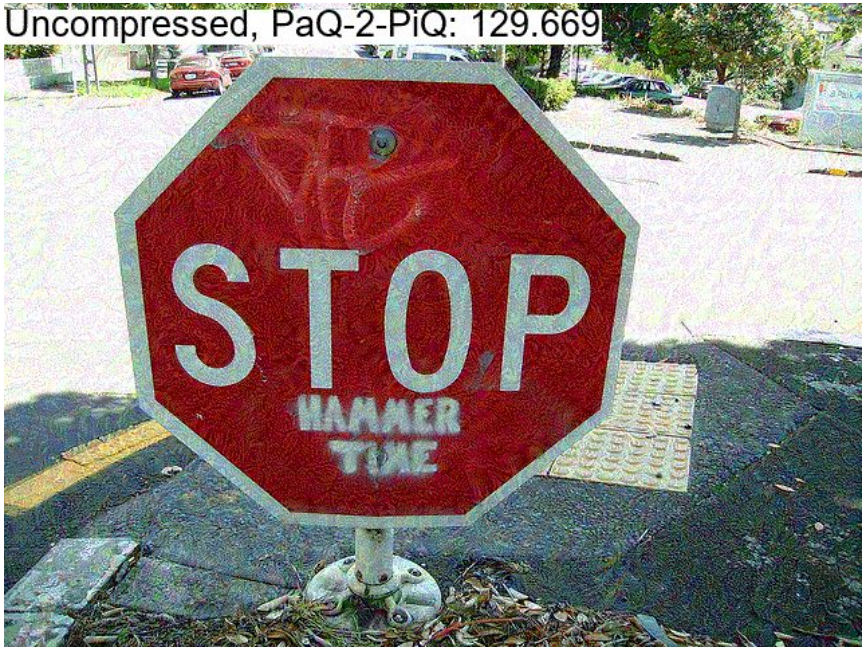
Задание по атакам на метрики

Пример графика для одного ε



Задание по атакам на метрики

Визуализация сжатия



Сжатие атакованного изображения

Сжатие исходного изображения

Задание по атакам на метрики

Оценивание (2)

- Для каждого изображения берется среднее арифметическое этих площадей.
- Итоговый результат получается как среднее арифметическое полученных для каждого изображения величин.

Задание по атакам на метрики

Идеи по улучшению результатов

- Поиграться с оптимизаторами и их параметрами (вычислительно трудно)
- Поиграться с лосс-функцией (придумать составную, можно постараться сделать атаку менее заметной)
- Добавить в обучение дифференцируемый JPEG

Задание по атакам на метрики

Структура задания

- test.py - скрипт для тестирования атаки
- iterative.py - файл решения
- uap.py - файл решения
- uap_train_script.py - вспомогательный скрипт для запуска обучения UAP добавки
- paq2piq_standalone.py - код для PaQ-2-PiQ. В файле объявлен класс MetricModel, реализующий метрику как torch.nn модуль с методом forward, поддерживающий backpropagation.
- weights/ - веса для метрики PaQ-2-PiQ
  - RoIPoolModel.pth
- training_dataset/ - обучающий датасет для UAP атаки (https://dione.gml-team.ru:5001/sharing/63K2wpfLJ)
- public_dataset/ - публичная часть датасета для тестов (https://dione.gml-team.ru:5001/sharing/fnB9HbuB9)

```
python uap_train_script.py --path_train ./training_dataset --save_path ./uap_paq2piq.png
--model_weights ./weights/RoIPoolModel.pth --batch_size 8 --device cuda:0

python test.py --attack_type uap --uap_train_path ./uap_paq2piq.png --csv_results_dir
results_dir --device cuda:0 --dataset_path ./public_dataset
--model_weights ./weights/RoIPoolModel.pth
```