



Stable Diffusion Laundering Attacks

Trends & Applications of Computer Vision

2024/2025

Martina D'Angelo, Thomas Lazzerini, Samuele Cappelletti



Our idea

- Explore more in depth the approaches proposed in Mandelli's paper^[A]:
 - Verify the **capabilities** of the SD-based laundering attack using the MMLAB dataset
 - Explore possible **methodologies** of SD-based laundering attack
 - Verify the **robustness** of the detector proposed in [A]

Recap: Image Laundering Problem



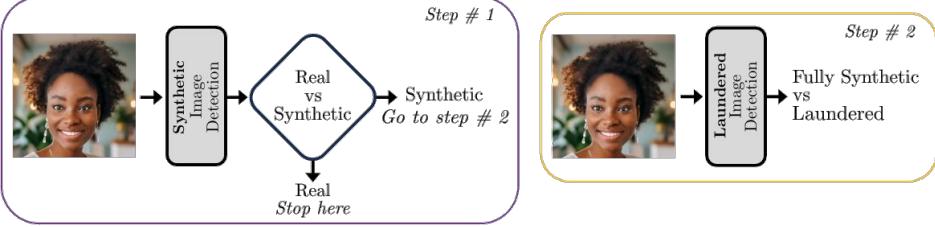
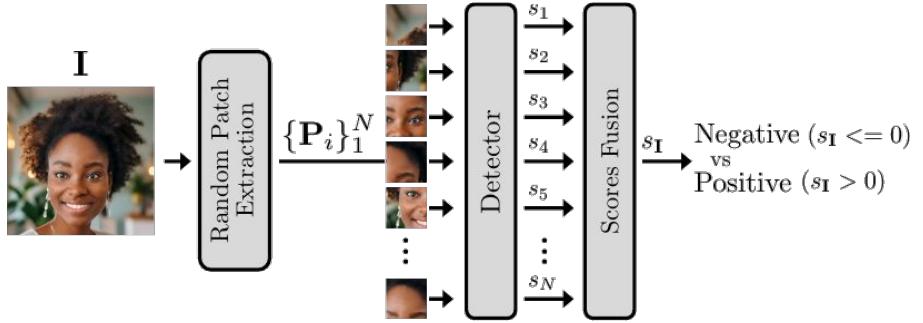
=====>
strength = 0



- What's happening:
 - The real image is encoded in the latent space using the SD encoder
 - Given the strength = 0, no noise is added and no denoising steps are performed
 - Then the the image is decoded using the SD decoder, introducing some **invisible artifacts**
- This is a problem:
 - the laundered image has no real forgery
 - despite this, it is classified as fake by many SoTA detectors
 - this approach could conceal nasty stuff as fake, despite being highly real

Recap: How To Address the Laundering Problem

- Problems with backbone:
 - architecture perform only **real vs. synthetic**
 - laundered images detected as fully synthetic
 - trained only on real and fully synthetic images
- Solution (Two Stage Architecture):
 - a first stage to discriminate **real images from all the others** (both synthetic and laundered)
 - a second stage to discriminate fully synthetic from laundered images





Our Dataset

(based on MMLAB's TrueFake)



Dataset

- The MMLAB dataset is composed by around 750K images:
 - Real Images
 - Synthetic images generated by different families of generators
 - Images generated at half precision (Float16)
- Due to computational limitations, we created our own MMLAB-based dataset:
 - 2.5K synthetic images generated by FLUX.1, SD2, SD3, SDXL, StyleGAN3
 - 1K real images
 - 18K laundered images



better results

visual prompt

visual prompt

the input image

reference steps



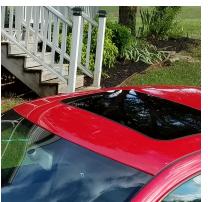
Our Approach to Laundering

- General parameters:

- StableDiffusionXL model
- Blank positive/negative prompt
- Guidance scale: 10
- Inference steps: 50
- Images 1024x1024 resolution

- Attack methodologies

- Multiple strength values [0.00, 0.05, 0.10, 0.15, 0.20]
- Multiple iterations on strength = 0 [passages 1-5]
- Different precision values [Float16, Float32]



Visual differences: multiple iterations



original



str=0, 1 iteration



str=0, 2 iterations



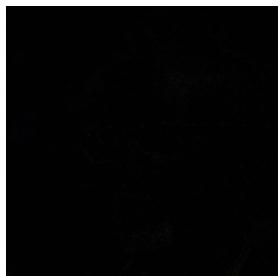
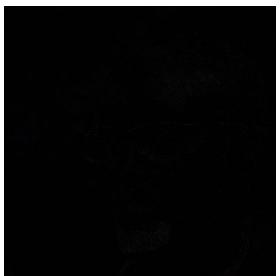
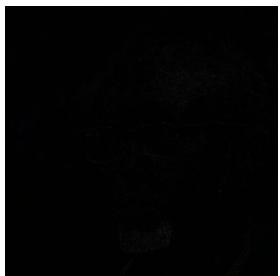
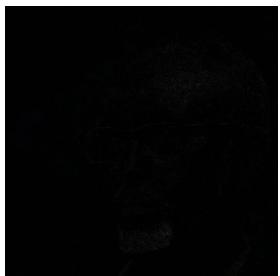
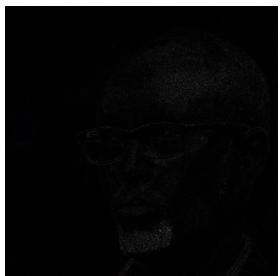
str=0, 3 iterations



str=0, 4 iterations



str=0, 5 iterations



Pixel level differences between current and previous iteration

Visual differences: multiple strengths



original



str=0



str=0.05



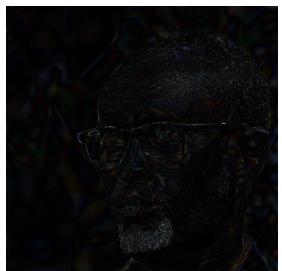
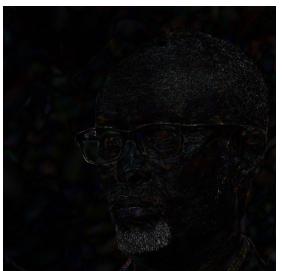
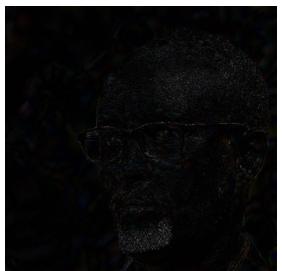
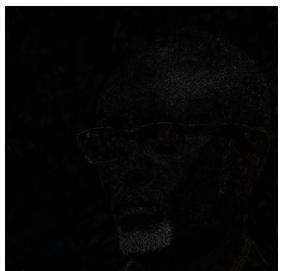
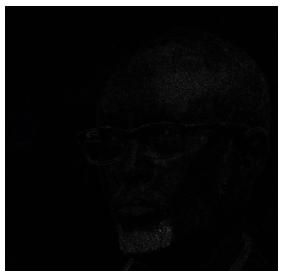
str=0.10



str=0.15



str=0.20



Pixel level differences between laundered and original image



Stable Diffusion UI

- The laundering through SD-based Hugging Face pipeline yielded an error:

```
RuntimeError: cannot reshape tensor of 0 elements into shape [0, -1, 1, 512]
because the unspecified dimension size -1 can be any value and is ambiguous
```

- To solve it, we used the web UI^[A] from *Automatic1111*:

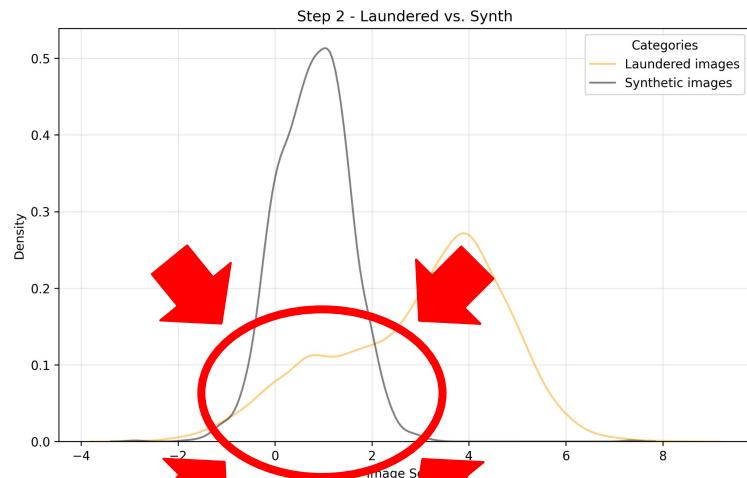
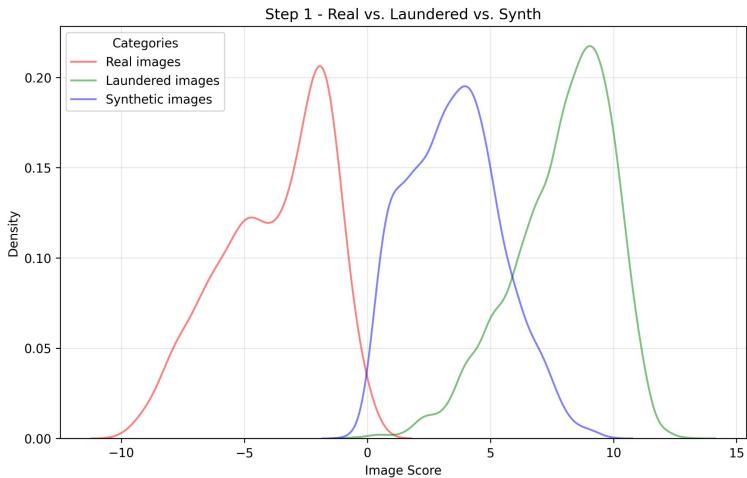
- not usable on MMLAB machine via CLI
- one of the most used tools in the SD community
- useful to impersonate a possible attacker
- usable only for Float32 generation on SDXL
- CLI parameters used: --xformers, --no-half, --medvram



Our Results

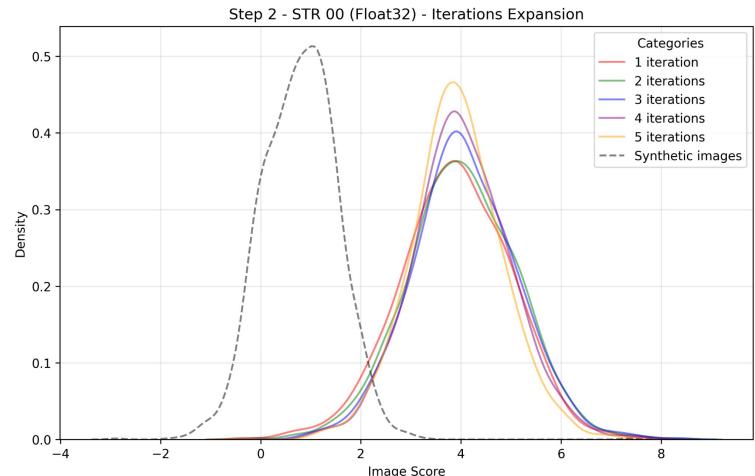
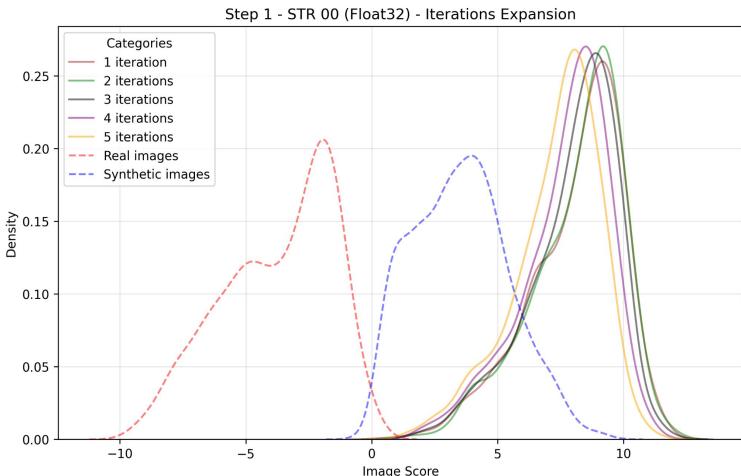
Results: general categories

- Over our categories Real, Laundered, (Fully) Synthetic:
 - generally coherent behaviour with results proposed in [A] for Step 1
 - important overlap between Laundered and Synthetic for Step 2



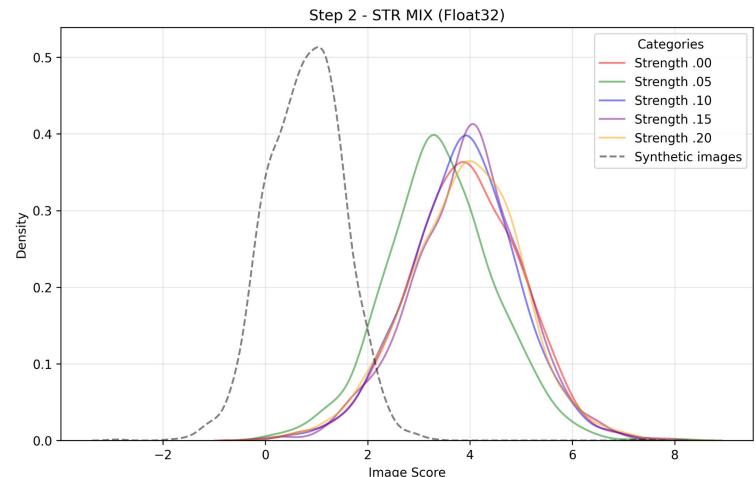
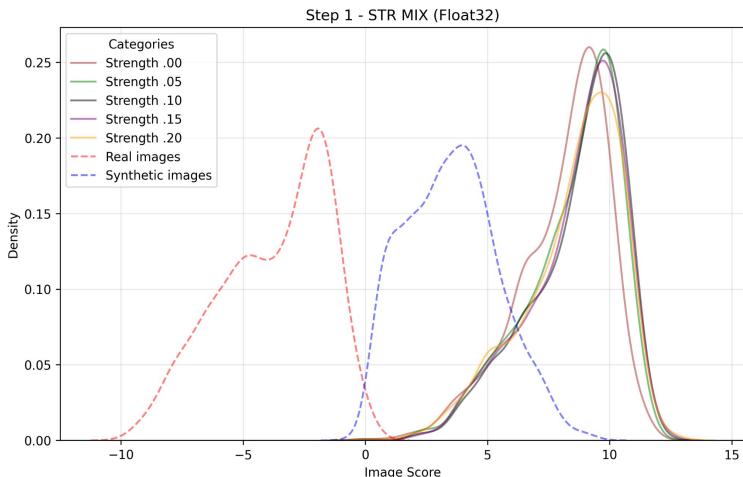
Results: strength = 0, multiple iterations

- Over multiple laundering iterations with strength 0 (precision Float32):
 - average score slightly decreasing over higher iteration values for Step 1
 - similar scores for Step 2



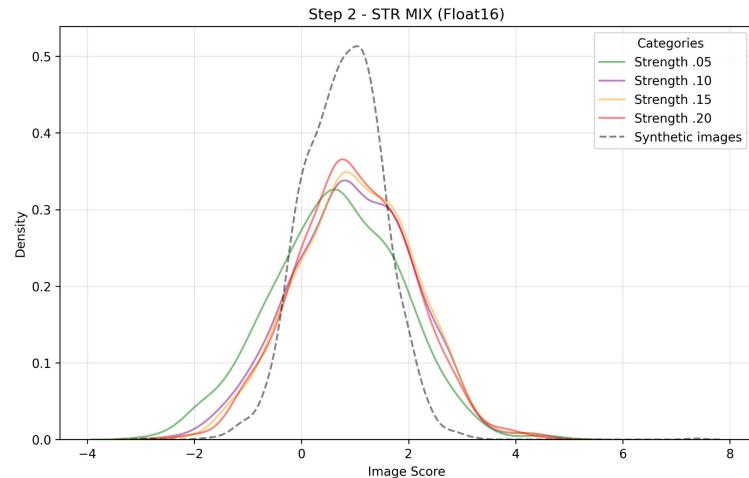
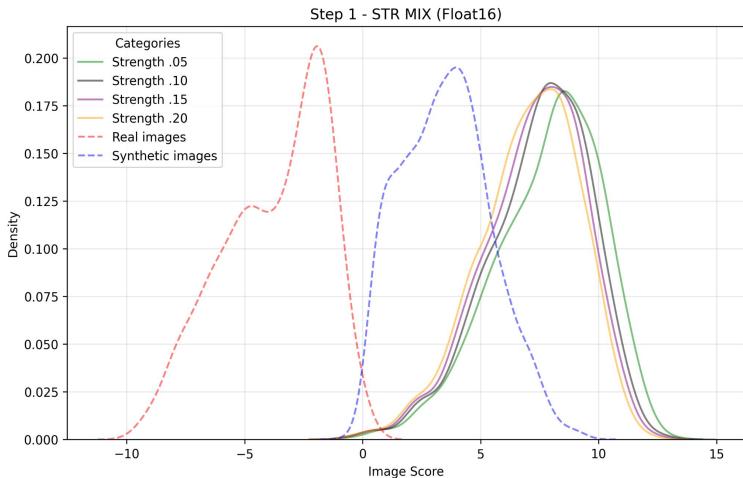
Results: different strength values

- Over laundering with multiple strength values (precision Float32):
 - no noticeable differences between Step 1 and Step 2



Results: precision

- Over laundering with multiple strength values (precision Float16):
 - no important differences in Step 1 compared to Float32
 - perfect overlap between fully synthetic and laundered images for all strength values!





Results: Float16 vs Float32

Float16, strength = 0.05

FAKE



Float32, strength = 0.05

LAUNDERED





Conclusion

- Multiple Float32 laundering iterations at strength = 0:
 - minor decrease of Step 1 score with higher iterations
- Float32 laundering at different strength values:
 - no visible results in both Step 1 and 2
- Float16 images:
 - both laundered and fully synthetic yields similar scores in Step 2, independently from the strength
 - We assume the same behaviour with strength = 0 laundering
 - This results suggest a possible bias in the Step 2 detector from [A]



Other open challenges

- Explore a possible integration of the Mimicry Attack^[A] with the SD-based laundering attack^[B]
- Solve the “strength = 0 laundering problem”
- Launder and test strength = 0 Float16 images
- Investigate possible biases in the detector from [B] between Float32 and Float16 images
- Verify possible effects of higher iteration numbers in the strength = 0 laundering



Thanks for the attention