2-face-similarity-thanos

January 16, 2024

[ ]:

```python
from pathlib import Path
import matplotlib.pyplot as plt


from lightweight_gan import LightweightGAN
import torch
import numpy as np

from PIL import Image
from torchvision.transforms import v2
```

[ ]:

```python
MODEL_PATH = Path("model/model_mercury_5_param_9000.pt")
LATENT_DIM = 256
IMG_SIZE = 256

torch.manual_seed(42)
```

[ ]:

```
<torch._C.Generator at 0x7fc01c28f210>
```

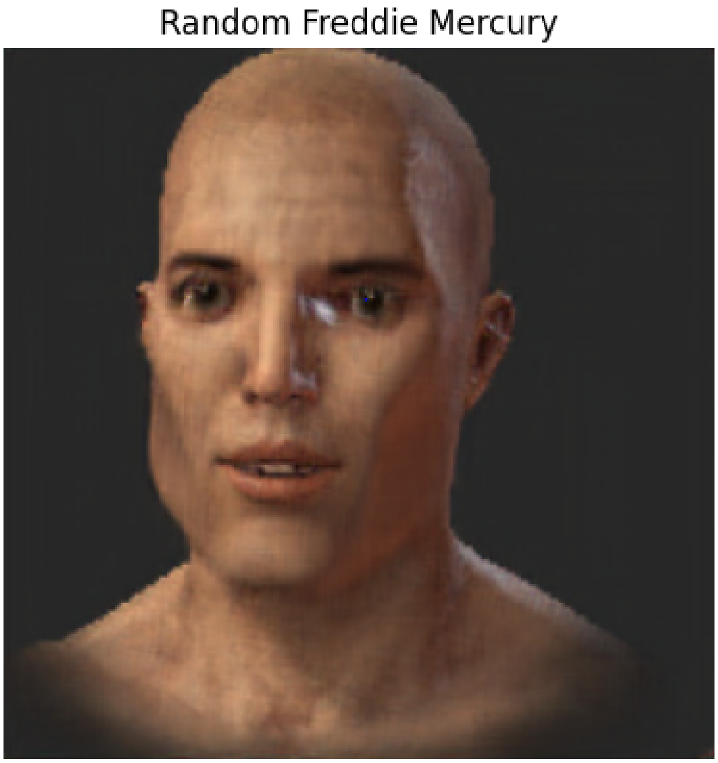
[ ]:

```python
def prep_tensor_to_show(tensor):
    return tensor.permute(1, 2, 0).detach().to('cpu').numpy()

def show_tensor_picture(tensor, title=""):
    im_arr = prep_tensor_to_show(tensor)
    plt.title(title)
    plt.axis("off")
    plt.imshow((im_arr*255).astype(np.uint8))


def import_generator(model_filepath: Path) -> torch.nn.Module:
    model = torch.load(model_filepath)

    gan = LightweightGAN(latent_dim=LATENT_DIM,
                         image_size=IMG_SIZE,
                         attn_res_layers=[32],
                         freq_chan_attn=False)
    gan.load_state_dict(model["GAN"], strict=False)
    gan.eval()
    return gan.G

def prep_image(filename)-> torch.tensor:
    image = Image.open(filename).convert("RGB")
    image_min_size = min(image.size)
    transforms = v2.Compose([
        v2.ToImage(),
        v2.CenterCrop(image_min_size),
        v2.Resize((IMG_SIZE, IMG_SIZE), antialias=True),
        v2.ToDtype(torch.float32, scale=True)
    ])
    return transforms(image).cuda(0)


def plot_images(images, labels, nrows, ncols, step=1, title="", figsize=[8, 4]):
    """Plot nrows x ncols images from images and set labels as titles."""
    fig, axes = plt.subplots(nrows=nrows, ncols=ncols, figsize=figsize)
    for i, ax in enumerate(axes.flat):
        idx = i * step
        if images[i].shape[-1] == 1:
            ax.imshow((255*images[i]).reshape(images[idx].shape[0], images[idx].
 ↪shape[1]).astype(np.uint8))
        else:
            ax.imshow((255*images[idx]).astype(np.uint8))
        ax.set_xticks([])
        ax.set_yticks([])
        ax.set_title(labels[idx])
    fig.suptitle(title)
    plt.tight_layout()
    plt.show()

def plot_loss(losses, iterations, title=""):
    plt.plot(iterations, losses)
    plt.title(title)
    plt.xlabel("Iteration")
    plt.ylabel("Loss")
    plt.ylim(bottom=0, top=1.1*max(losses))
    plt.show()


def present_results(z, checkpoints, loss, original_image_path):
    plot_images([checkpoint for _, checkpoint in checkpoints],
            [f"iteration {checkpoint_n}" for checkpoint_n, _ in checkpoints],
            nrows=len(checkpoints)//4, ncols=4,
            title="Generated images for parameters with value 128", figsize=[8,␣
 ↪6])
```

```
    original_middle_image = prep_tensor_to_show(prep_image("img/from_dataset/
↪middle_mercury.png"))
    inffered_middle_image = prep_tensor_to_show(generator(z)[0])
    plot_images([original_middle_image, inffered_middle_image],
                ["original", "inferred"],
                nrows=1, ncols=2,
                title="Original and inferred image",
                figsize=[8, 4])
```

[ ]:

```
generator = import_generator(MODEL_PATH)

random_latent_vectors = torch.randn( (1, LATENT_DIM) ).cuda(0)
generated_images = generator(random_latent_vectors)
show_tensor_picture(generated_images[0], title="Random Freddie Mercury")
```

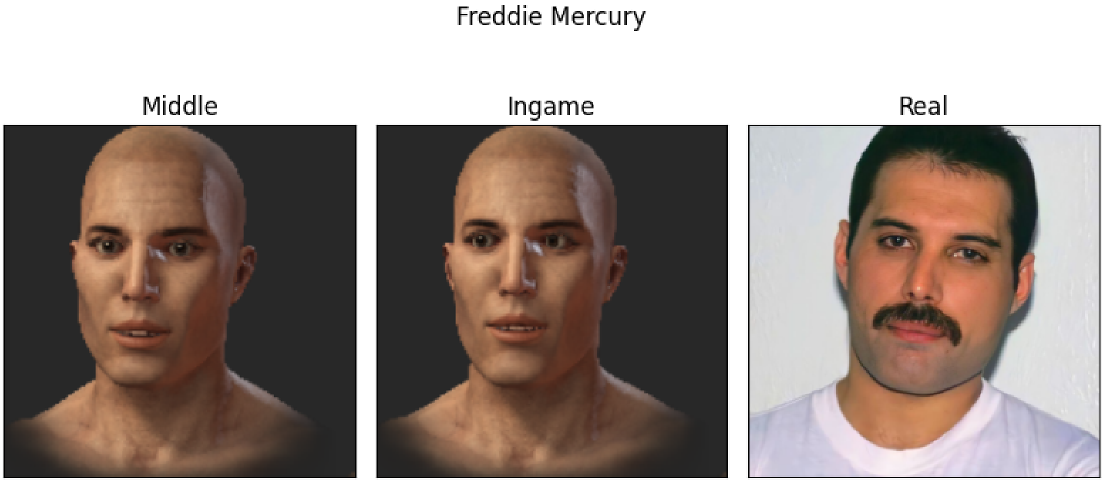


[ ]:

```
def present_results(original_image, z, checkpoints, losses):
    plot_images([checkpoint for _, checkpoint in checkpoints],
                [f"iteration {checkpoint_n}" for checkpoint_n, _ in checkpoints],
                nrows=len(checkpoints)//4, ncols=4,
                title="Generated images for parameters with value 128", figsize=[8,␣
↪6])
```

```
    original_image = prep_tensor_to_show(original_image)
    inffered_image = prep_tensor_to_show(generator(z)[0])
    plot_images([original_image, inffered_image],
        ["original", "inferred"],
        nrows=1, ncols=2,
        title="Original and inferred image",
        figsize=[8, 4])

    plot_loss(losses, [i for i, _ in checkpoints], title="Loss")
```

```
[ ]: MIDDLE_IMAGE_PATH = "img/from_dataset/middle_mercury.png"
FREDDIE_INGAME_PATH = "img/from_dataset/perfect_mercury.png"
FREDDIE_REAL_PATH = "img/real-freddies/1.jpg"

middle_image = prep_image(MIDDLE_IMAGE_PATH)
freddie_ingame = prep_image(FREDDIE_INGAME_PATH)
freddie_real = prep_image(FREDDIE_REAL_PATH)

plot_images([prep_tensor_to_show(img) for img in (middle_image, freddie_ingame,␣
 ↪freddie_real)],
      ["Middle", "Ingame", "Real"],
      1, 3, title="Freddie Mercury")
```



0.1 MSE

0.1.1 Middle picture

[ ]:

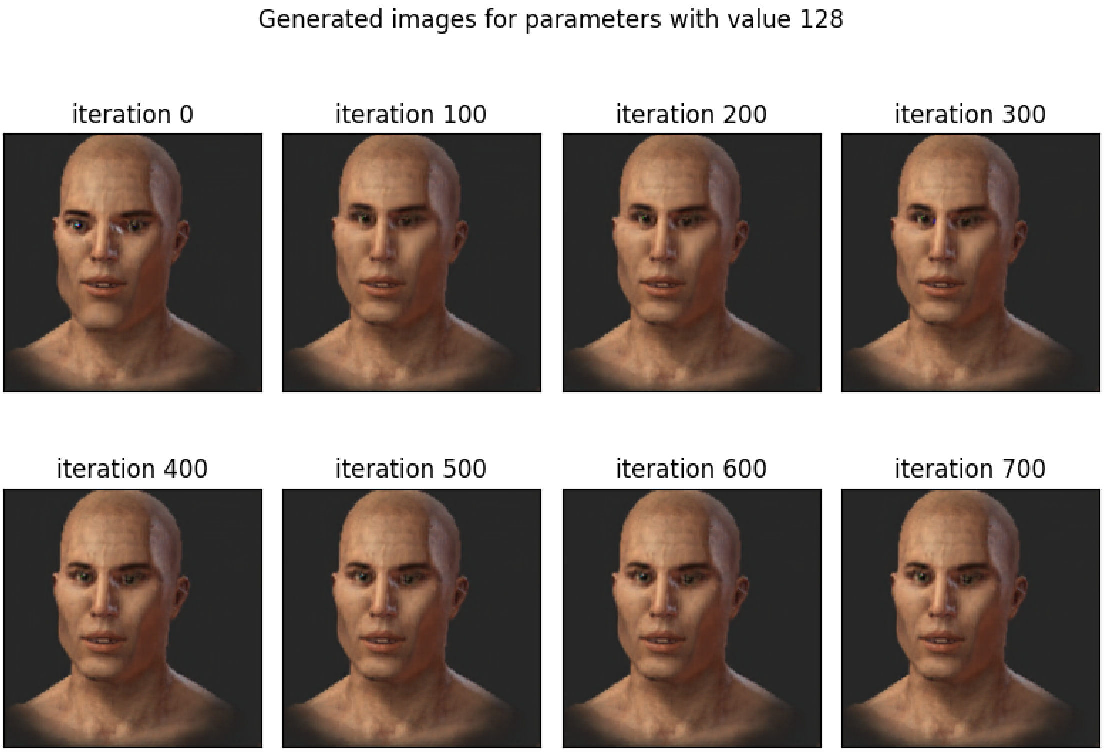
```
## MSE

def find_z_MSE(exemplary_photo, iterations=100, lr=5, checkpoint_step=10,␣
 ↪z_init=None):
    z = torch.randn( (1, LATENT_DIM), requires_grad=True, device="cuda") \
        if z_init is None else z_init
    y_target = exemplary_photo

    criterion = torch.nn.MSELoss()
    optimizer = torch.optim.Adam([z], lr=lr)

    checkpoints = []
    losses = []
    for i in range(iterations):
        y_pred = generator(z)[0]
        loss = criterion(y_target, y_pred)

        if i % checkpoint_step == 0:
            checkpoints.append((i, y_pred.detach().permute((1,2,0)).cpu().
 ↪numpy()))
            losses.append(loss.item())
            print(f"Iteration: {i}, loss: {loss.item():.2E}")

        optimizer.zero_grad()
        loss.backward()
        optimizer.step()

    return z, checkpoints, losses
```

[ ]:

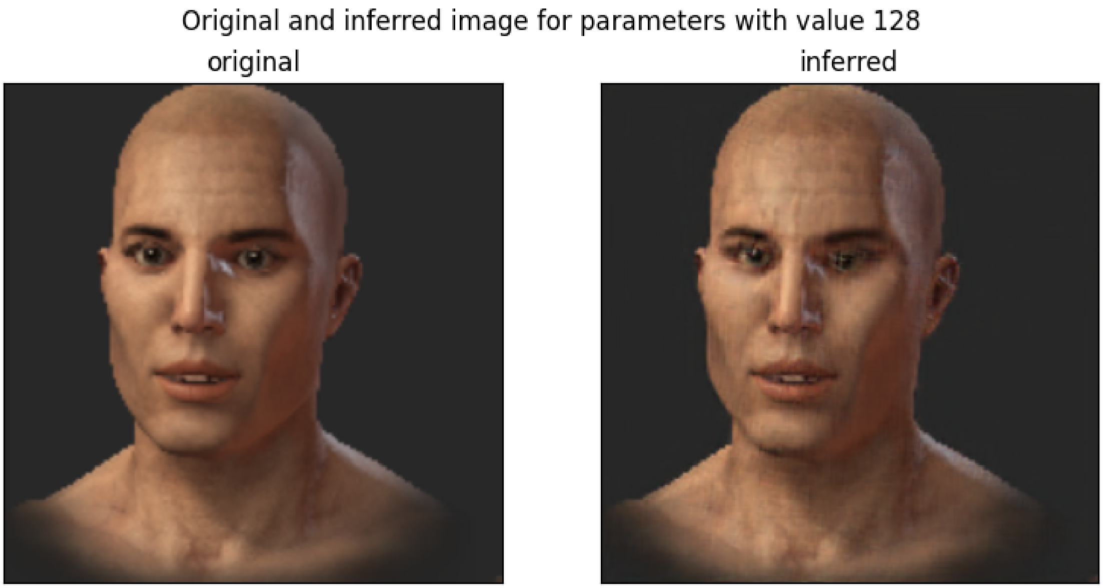
```
middle_z, middle_checkpoints, middle_losses = find_z_MSE(middle_image,␣
 ↪iterations=1000, lr=1, checkpoint_step=100)
```

```
Iteration: 0, loss: 1.14E-03
Iteration: 100, loss: 7.30E-04
Iteration: 200, loss: 6.83E-04
Iteration: 300, loss: 6.32E-04
Iteration: 400, loss: 4.40E-04
Iteration: 500, loss: 4.25E-04
Iteration: 600, loss: 4.15E-04
Iteration: 700, loss: 4.07E-04
Iteration: 800, loss: 4.00E-04
Iteration: 900, loss: 3.92E-04
```

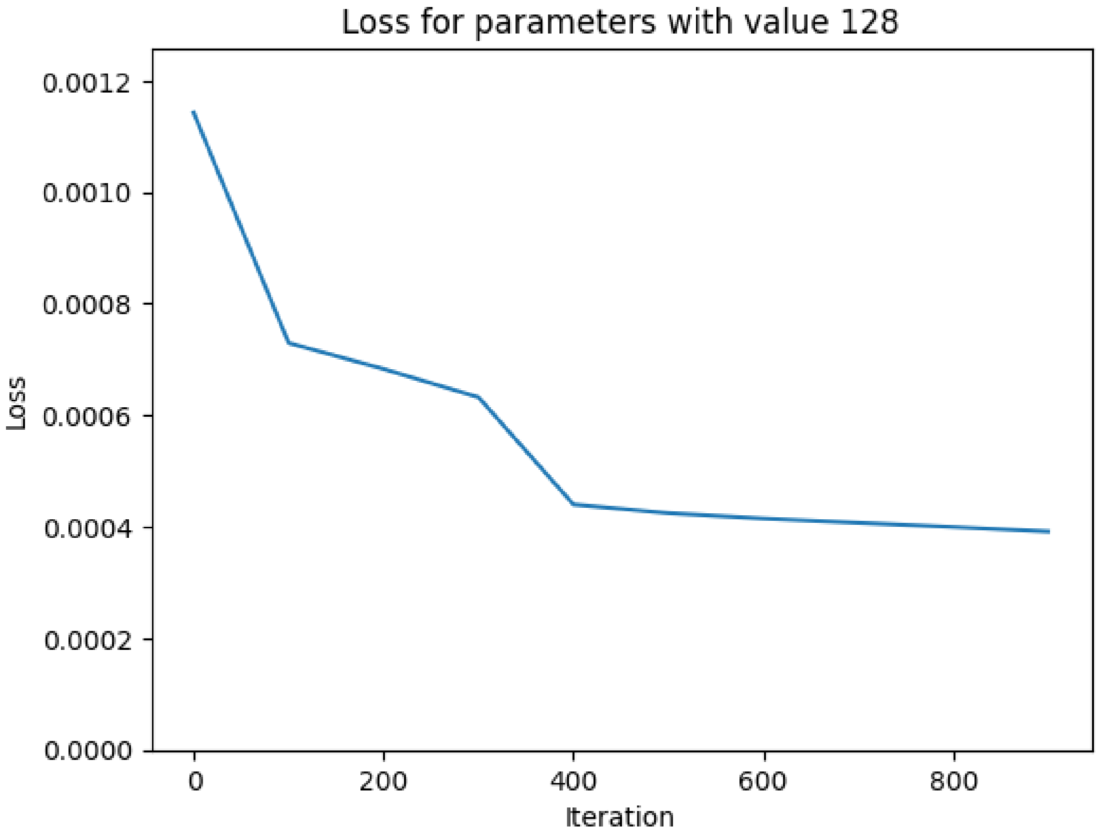
```
[ ]: plot_images([checkpoint for _, checkpoint in middle_checkpoints],
                 [f"iteration {checkpoint_n}" for checkpoint_n, _ in␣
      ↪middle_checkpoints],
                 nrows=len(middle_checkpoints)//4, ncols=4,
                 title="Generated images for parameters with value 128", figsize=[8,␣
      ↪6])
```



```
[ ]: original_middle_image = prep_tensor_to_show(prep_image("img/from_dataset/
      ↪middle_mercury.png"))
     inffered_middle_image = prep_tensor_to_show(generator(middle_z)[0])

     plot_images([original_middle_image, inffered_middle_image], ["original",␣
      ↪"inferred"], nrows=1, ncols=2, title="Original and inferred image for␣
      ↪parameters with value 128", figsize=[8, 4])
```



```
[ ]: plot_loss(middle_losses, [i for i, _ in middle_checkpoints], title="Loss for parameters with value 128")
```

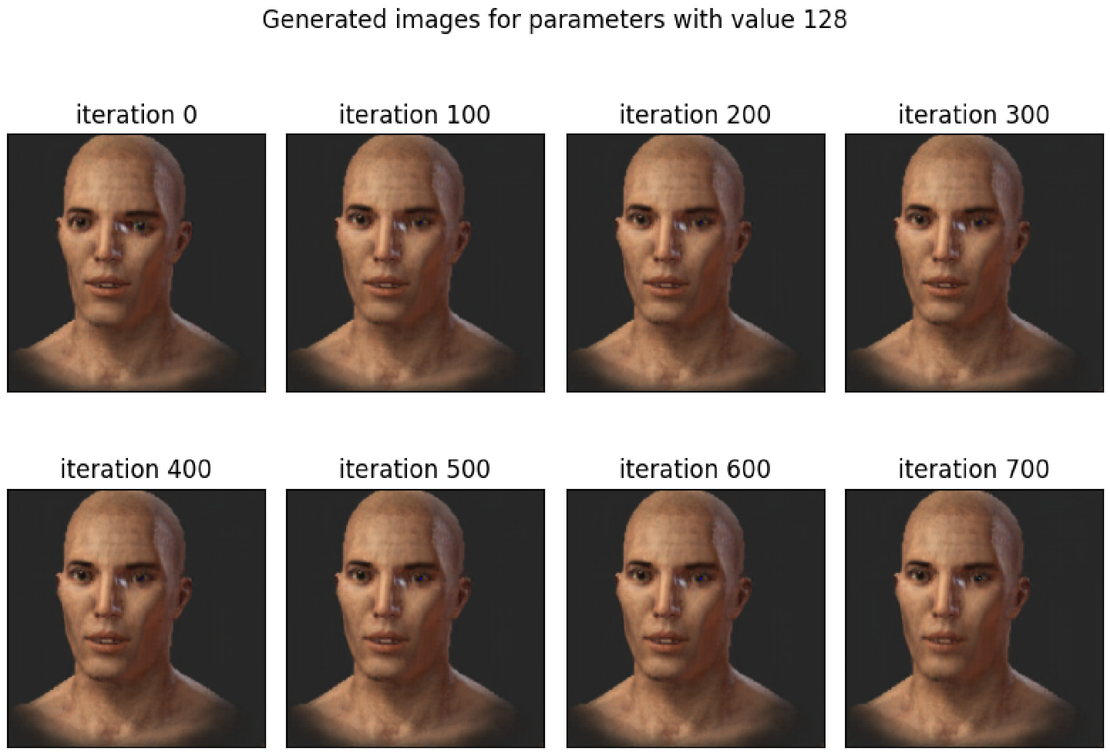


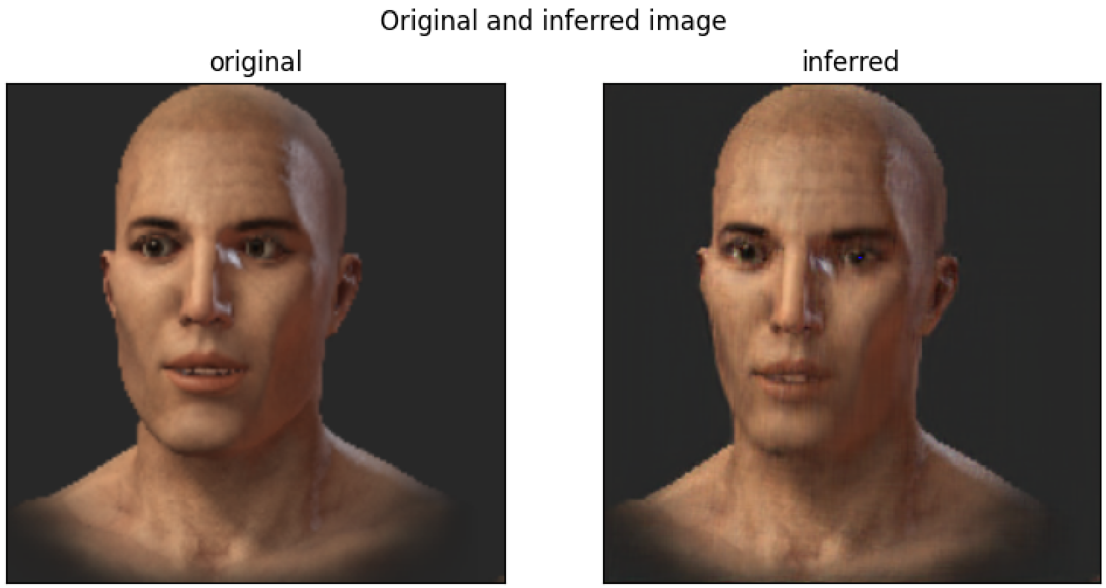
0.2 Ingame freddie

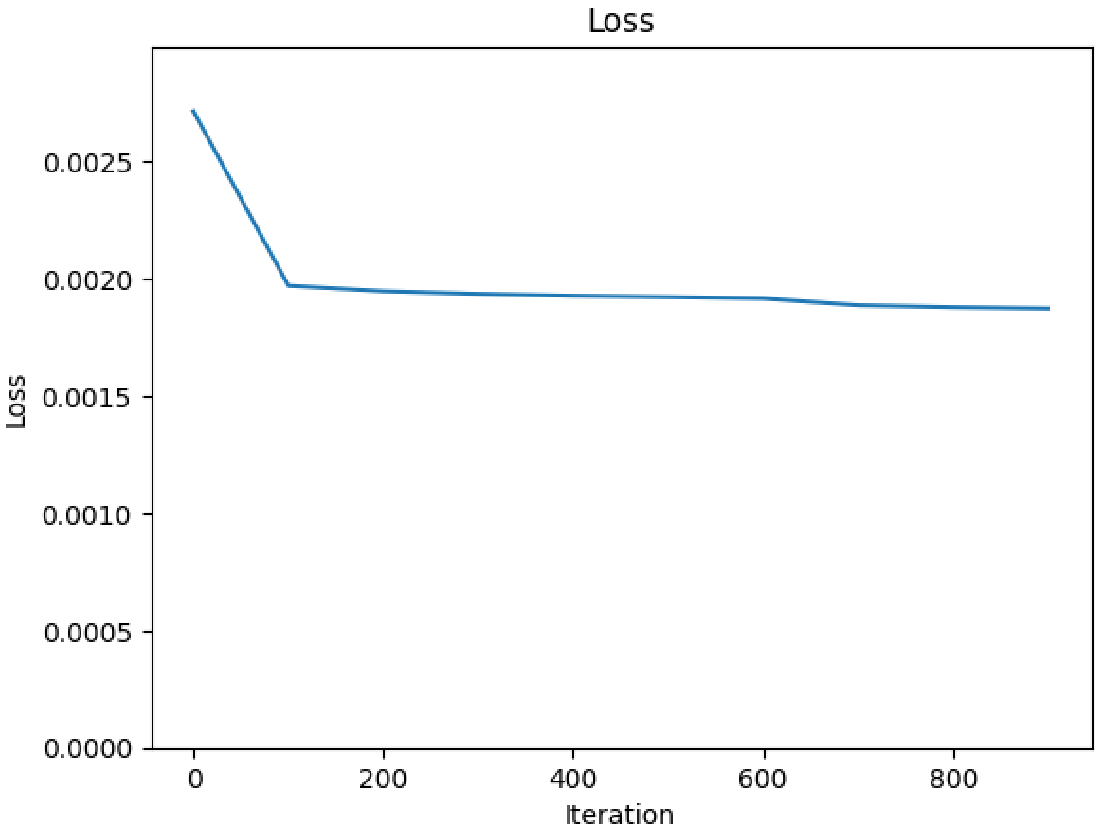
```
[ ]: ingame_freddie_z_fit = find_z_MSE(freddie_ingame, iterations=1000, lr=1,␣
      ↪checkpoint_step=100)
```

```
Iteration: 0, loss: 2.71E-03
Iteration: 100, loss: 1.97E-03
Iteration: 200, loss: 1.95E-03
Iteration: 300, loss: 1.94E-03
Iteration: 400, loss: 1.93E-03
Iteration: 500, loss: 1.92E-03
Iteration: 600, loss: 1.92E-03
Iteration: 700, loss: 1.89E-03
Iteration: 800, loss: 1.88E-03
Iteration: 900, loss: 1.87E-03
```

```
[ ]: present_results(freddie_ingame, *ingame_freddie_z_fit)
```




Original and inferred image
original
inferred


Loss
Loss
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
0
200
400
600
800
Iteration

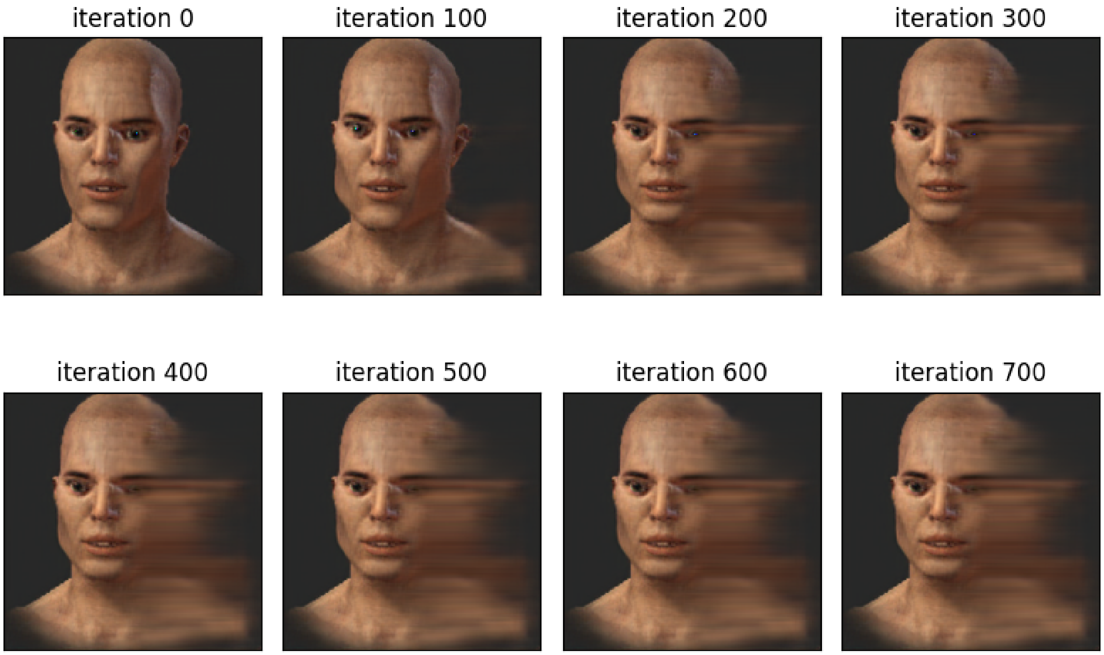
0.2.1 Real Freddie

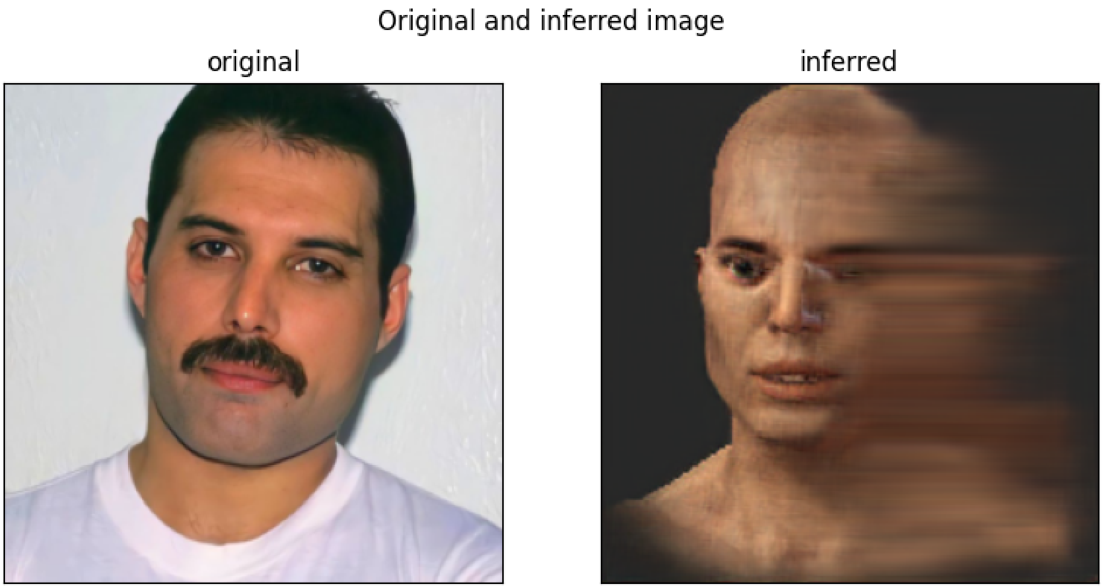
```
[ ]: real_freddie_z_fit = find_z_MSE(freddie_real, iterations=1000, lr=1,␣
      ↪checkpoint_step=100)
```

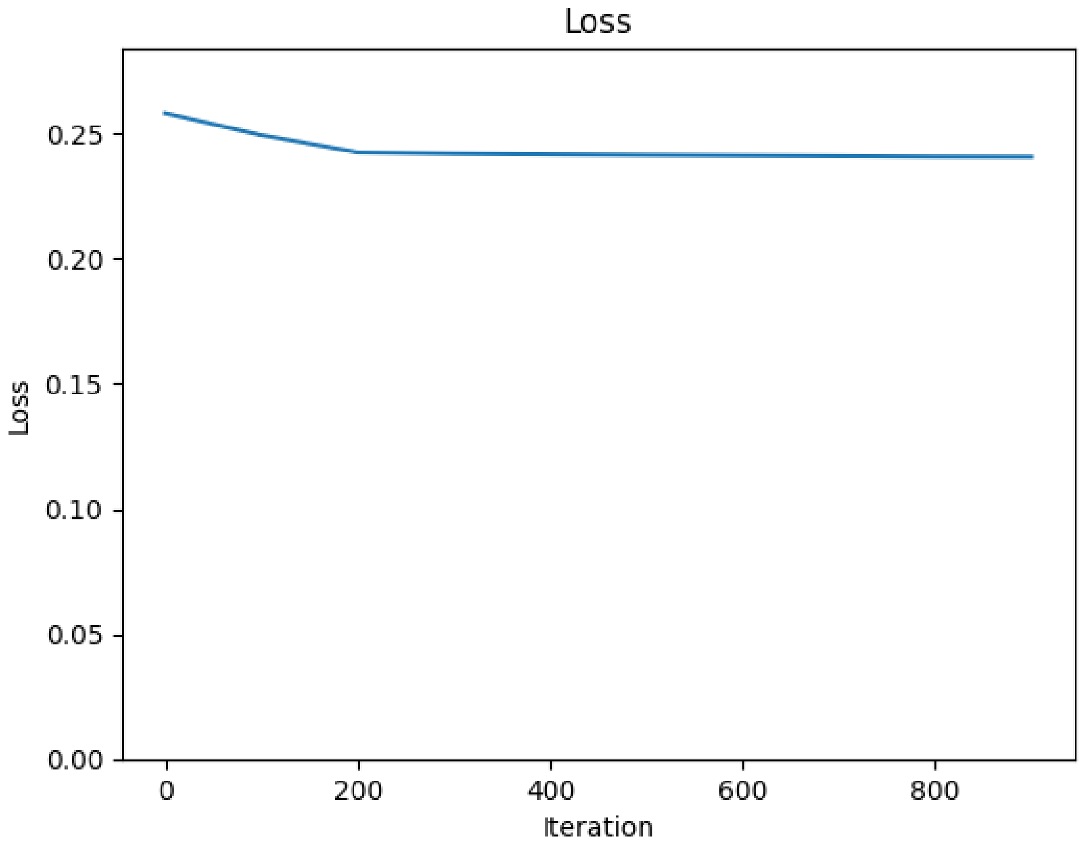
```
Iteration: 0, loss: 2.58E-01
Iteration: 100, loss: 2.49E-01
Iteration: 200, loss: 2.42E-01
Iteration: 300, loss: 2.42E-01
Iteration: 400, loss: 2.42E-01
Iteration: 500, loss: 2.41E-01
Iteration: 600, loss: 2.41E-01
Iteration: 700, loss: 2.41E-01
Iteration: 800, loss: 2.41E-01
Iteration: 900, loss: 2.41E-01
```

```
[ ]: present_results(freddie_real, *real_freddie_z_fit)
```

Generated images for parameters with value 128




Original and inferred image
original
inferred


Loss
0.25
0.20
0.15
0.10
0.05
0.00
Loss
0
200
400
600
800
Iteration

0.3 FaceNet512

Uzyskał zdecydowanie najlepsze wyniki w porównywaniu twarzy Freddiego z grą. Czy uda mu się znaleźć odpowiedni wektor z przestrzeni ukrytej?

0.3.1 Czy on w ogóle wykrywa twarz z gry?

[ ]:
```
from deepface import DeepFace
```

```
2024-01-16 22:14:45.758080: E
external/local_xla/xla/stream_executor/cuda/cuda_dnn.cc:9261] Unable to register
cuDNN factory: Attempting to register factory for plugin cuDNN when one has
already been registered
2024-01-16 22:14:45.758114: E
external/local_xla/xla/stream_executor/cuda/cuda_fft.cc:607] Unable to register
cuFFT factory: Attempting to register factory for plugin cuFFT when one has
already been registered
2024-01-16 22:14:45.759268: E
external/local_xla/xla/stream_executor/cuda/cuda_blas.cc:1515] Unable to
register cuBLAS factory: Attempting to register factory for plugin cuBLAS when
one has already been registered
2024-01-16 22:14:45.765986: I tensorflow/core/platform/cpu_feature_guard.cc:182]
This TensorFlow binary is optimized to use available CPU instructions in
performance-critical operations.
To enable the following instructions: AVX2 FMA, in other operations, rebuild
TensorFlow with the appropriate compiler flags.
2024-01-16 22:14:46.572157: W
tensorflow/compiler/tf2tensorrt/utils/py_utils.cc:38] TF-TRT Warning: Could not
find TensorRT
```

```
24-01-16 22:14:47 - Directory /home/pawel/.deepface created
24-01-16 22:14:47 - Directory /home/pawel/.deepface/weights created
```

[ ]:
```
DeepFace.find(img_path=FREDDIE_REAL_PATH,
              db_path="img/from_dataset",
              model_name="Facenet512",
              distance_metric="euclidean_l2")
```

```
Finding representations:   0%|          | 0/17 [00:00<?, ?it/s]2024-01-16
22:17:33.637138: I
external/local_xla/xla/stream_executor/cuda/cuda_executor.cc:901] successful
NUMA node read from SysFS had negative value (-1), but there must be at least
one NUMA node, so returning NUMA node zero. See more at
https://github.com/torvalds/linux/blob/v6.0/Documentation/ABI/testing/sysfs-bus-
pci#L344-L355
2024-01-16 22:17:33.642013: I
external/local_xla/xla/stream_executor/cuda/cuda_executor.cc:901] successful
NUMA node read from SysFS had negative value (-1), but there must be at least
one NUMA node, so returning NUMA node zero. See more at
https://github.com/torvalds/linux/blob/v6.0/Documentation/ABI/testing/sysfs-bus-
```

```
pci#L344-L355
2024-01-16 22:17:33.642173: I
external/local_xla/xla/stream_executor/cuda/cuda_executor.cc:901] successful
NUMA node read from SysFS had negative value (-1), but there must be at least
one NUMA node, so returning NUMA node zero. See more at
https://github.com/torvalds/linux/blob/v6.0/Documentation/ABI/testing/sysfs-bus-
pci#L344-L355
2024-01-16 22:17:33.643473: I
external/local_xla/xla/stream_executor/cuda/cuda_executor.cc:901] successful
NUMA node read from SysFS had negative value (-1), but there must be at least
one NUMA node, so returning NUMA node zero. See more at
https://github.com/torvalds/linux/blob/v6.0/Documentation/ABI/testing/sysfs-bus-
pci#L344-L355
2024-01-16 22:17:33.643612: I
external/local_xla/xla/stream_executor/cuda/cuda_executor.cc:901] successful
NUMA node read from SysFS had negative value (-1), but there must be at least
one NUMA node, so returning NUMA node zero. See more at
https://github.com/torvalds/linux/blob/v6.0/Documentation/ABI/testing/sysfs-bus-
pci#L344-L355
2024-01-16 22:17:33.643735: I
external/local_xla/xla/stream_executor/cuda/cuda_executor.cc:901] successful
NUMA node read from SysFS had negative value (-1), but there must be at least
one NUMA node, so returning NUMA node zero. See more at
https://github.com/torvalds/linux/blob/v6.0/Documentation/ABI/testing/sysfs-bus-
pci#L344-L355
2024-01-16 22:17:34.728925: I
external/local_xla/xla/stream_executor/cuda/cuda_executor.cc:901] successful
NUMA node read from SysFS had negative value (-1), but there must be at least
one NUMA node, so returning NUMA node zero. See more at
https://github.com/torvalds/linux/blob/v6.0/Documentation/ABI/testing/sysfs-bus-
pci#L344-L355
2024-01-16 22:17:34.729112: I
external/local_xla/xla/stream_executor/cuda/cuda_executor.cc:901] successful
NUMA node read from SysFS had negative value (-1), but there must be at least
one NUMA node, so returning NUMA node zero. See more at
https://github.com/torvalds/linux/blob/v6.0/Documentation/ABI/testing/sysfs-bus-
pci#L344-L355
2024-01-16 22:17:34.729298: I
external/local_xla/xla/stream_executor/cuda/cuda_executor.cc:901] successful
NUMA node read from SysFS had negative value (-1), but there must be at least
one NUMA node, so returning NUMA node zero. See more at
https://github.com/torvalds/linux/blob/v6.0/Documentation/ABI/testing/sysfs-bus-
pci#L344-L355
2024-01-16 22:17:34.729401: I
tensorflow/core/common_runtime/gpu/gpu_device.cc:1929] Created device
/job:localhost/replica:0/task:0/device:GPU:0 with 1809 MB memory:  -> device: 0,
name: NVIDIA GeForce GTX 1050, pci bus id: 0000:01:00.0, compute capability: 6.1
```

```
24-01-16 22:17:36 - facenet512_weights.h5 will be downloaded…

Downloading…
From: https://github.com/serengil/deepface_models/releases/download/v1.0/facenet
512_weights.h5
To: /home/pawel/.deepface/weights/facenet512_weights.h5
100%|      | 95.0M/95.0M [00:07<00:00, 12.1MB/s]
2024-01-16 22:17:46.453841: I
external/local_xla/xla/stream_executor/cuda/cuda_dnn.cc:454] Loaded cuDNN
version 8902
2024-01-16 22:17:46.615104: W
external/local_xla/xla/service/gpu/llvm_gpu_backend/gpu_backend_lib.cc:504]
Can't find libdevice directory ${CUDA_DIR}/nvvm/libdevice. This may result in
compilation or runtime failures, if the program we try to run uses routines from
libdevice.
Searched for CUDA in the following directories:
  ./cuda_sdk_lib
  /usr/local/cuda-12.2
  /usr/local/cuda
  /home/pawel/Documents/studia/magisterka/CKTinderator/venv/lib/python3.11/site-
packages/tensorflow/python/platform/../../../nvidia/cuda_nvcc
  /home/pawel/Documents/studia/magisterka/CKTinderator/venv/lib/python3.11/site-
packages/tensorflow/python/platform/../../../../nvidia/cuda_nvcc
  .
You can choose the search directory by setting xla_gpu_cuda_data_dir in
HloModule's DebugOptions.  For most apps, setting the environment variable
XLA_FLAGS=--xla_gpu_cuda_data_dir=/path/to/cuda will work.
2024-01-16 22:17:48.175066: W
external/local_xla/xla/service/gpu/llvm_gpu_backend/gpu_backend_lib.cc:542]
libdevice is required by this HLO module but was not found at ./libdevice.10.bc
error: libdevice not found at ./libdevice.10.bc
2024-01-16 22:17:48.175279: E
tensorflow/compiler/mlir/tools/kernel_gen/tf_framework_c_interface.cc:207]
INTERNAL: Generating device code failed.
Finding representations:   0%|          | 0/17 [00:14<?, ?it/s]2024-01-16
22:17:48.176169: W tensorflow/core/framework/op_kernel.cc:1827] UNKNOWN: JIT
compilation failed.
```

```
---------------------------------------------------------------------------
UnknownError                              Traceback (most recent call last)
Cell In[18], line 1
----> 1 DeepFace.find(img_path=FREDDIE_REAL_PATH,
      2               db_path="img/from_dataset",
      3               model_name="Facenet512",
      4               distance_metric="euclidean_l2")
```

```
File ~/Documents/studia/magisterka/CKTinderator/venv/lib/python3.11/
 ↪site-packages/deepface/DeepFace.py:547, in find(img_path, db_path, model_name ␣
 ↪distance_metric, enforce_detection, detector_backend, align, normalization,␣
 ↪silent)
    537 img_objs = functions.extract_faces(
    538     img=employee,
    539     target_size=target_size,
   (…)
    543     align=align,
    544 )
    546 for img_content, img_region, _ in img_objs:
--> 547     embedding_obj = represent(
    548         img_path=img_content,
    549         model_name=model_name,
    550         enforce_detection=enforce_detection,
    551         detector_backend="skip",
    552         align=align,
    553         normalization=normalization,
    554     )
    556     img_representation = embedding_obj[0]["embedding"]
    558     instance = []

File ~/Documents/studia/magisterka/CKTinderator/venv/lib/python3.11/
 ↪site-packages/deepface/DeepFace.py:760, in represent(img_path, model_name,␣
 ↪enforce_detection, detector_backend, align, normalization)
    755 # represent
    756 # if "keras" in str(type(model)):
    757 if isinstance(model, Model):
    758     # model.predict causes memory issue when it is called in a for loop
    759     # embedding = model.predict(img, verbose=0)[0].tolist()
--> 760     embedding = model(img, training=False).numpy()[0].tolist()
    761     # if you still get verbose logging. try call
    762     # - `tf.keras.utils.disable_interactive_logging()`
    763     # in your main program
    764 else:
    765     # SFace and Dlib are not keras models and no verbose arguments
    766     embedding = model.predict(img)[0].tolist()

File ~/Documents/studia/magisterka/CKTinderator/venv/lib/python3.11/
 ↪site-packages/keras/src/utils/traceback_utils.py:70, in filter_traceback.
 ↪<locals>.error_handler(*args, **kwargs)
     67     filtered_tb = _process_traceback_frames(e.__traceback__)
     68     # To get the full stack trace, call:
     69     # `tf.debugging.disable_traceback_filtering()`
---> 70     raise e.with_traceback(filtered_tb) from None
     71 finally:
     72     del filtered_tb
```

```
File ~/Documents/studia/magisterka/CKTinderator/venv/lib/python3.11/
 ↪site-packages/tensorflow/python/framework/ops.py:5883, in␣
 ↪raise_from_not_ok_status(e, name)
   5881 def raise_from_not_ok_status(e, name) -> NoReturn:
   5882   e.message += (" name: " + str(name if name is not None else ""))
-> 5883   raise core._status_to_exception(e) from None

UnknownError: Exception encountered when calling layer 'Bottleneck_BatchNorm'␣
 ↪(type BatchNormalization).

{{function_node __wrapped__Rsqrt_device_/job:localhost/replica:0/task:0/device:
 ↪GPU:0}} JIT compilation failed. [Op:Rsqrt] name:

Call arguments received by layer 'Bottleneck_BatchNorm' (type␣
 ↪BatchNormalization):
  • inputs=tf.Tensor(shape=(1, 512), dtype=float32)
  • training=False
  • mask=None
```

```
[ ]:
```