Transformation-based Image Generation

Yaroslav Ganin

Intro

- Given two images $\mathbf{x}_s$ and $\mathbf{x}_g$, we would like to build a sequence of intermediate $\mathbf{x}_1$, $\mathbf{x}_2$, …, $\mathbf{x}_n$ connecting $\mathbf{x}_s$ and $\mathbf{x}_g$ (**interpolate**)
- Typically, one trains an **auto-encoder**-like model:
  - Map $\mathbf{x}_s$ and $\mathbf{x}_g$ into $\mathbf{z}_s$ and $\mathbf{z}_g$
  - Linearly interpolate between $\mathbf{z}_s$ and $\mathbf{z}_g$
  - Decode intermediate **z**s
- **Loss of information** as encoding and decoding are both lossy
- Could we interpolate in a more **direct** way?

Model

- Let $\mathbf{x}_S$ (images) be our **state space**: $\mathbb{S} = [0, 1]^{H \times W \times C}$
- The agent produces **actions** from $\mathbb{A}(\mathbf{x}) = \mathbb{R}^d$ (ideally we'd like to consider discrete actions as well; the agent should be able to decide when to stop the episode)
- Transition is performed by the **transformer**:

$$\mathbf{x}' = \mathbf{T}(\mathbf{x}, \mathbf{a})$$

- We seek to find a family of **policies** $\pi(\mathbf{x}; \mathbf{x}_g)$ bringing the agent from any $\mathbf{x}_s$ to any given $\mathbf{x}_g$
- Maximize expected discounted rewards

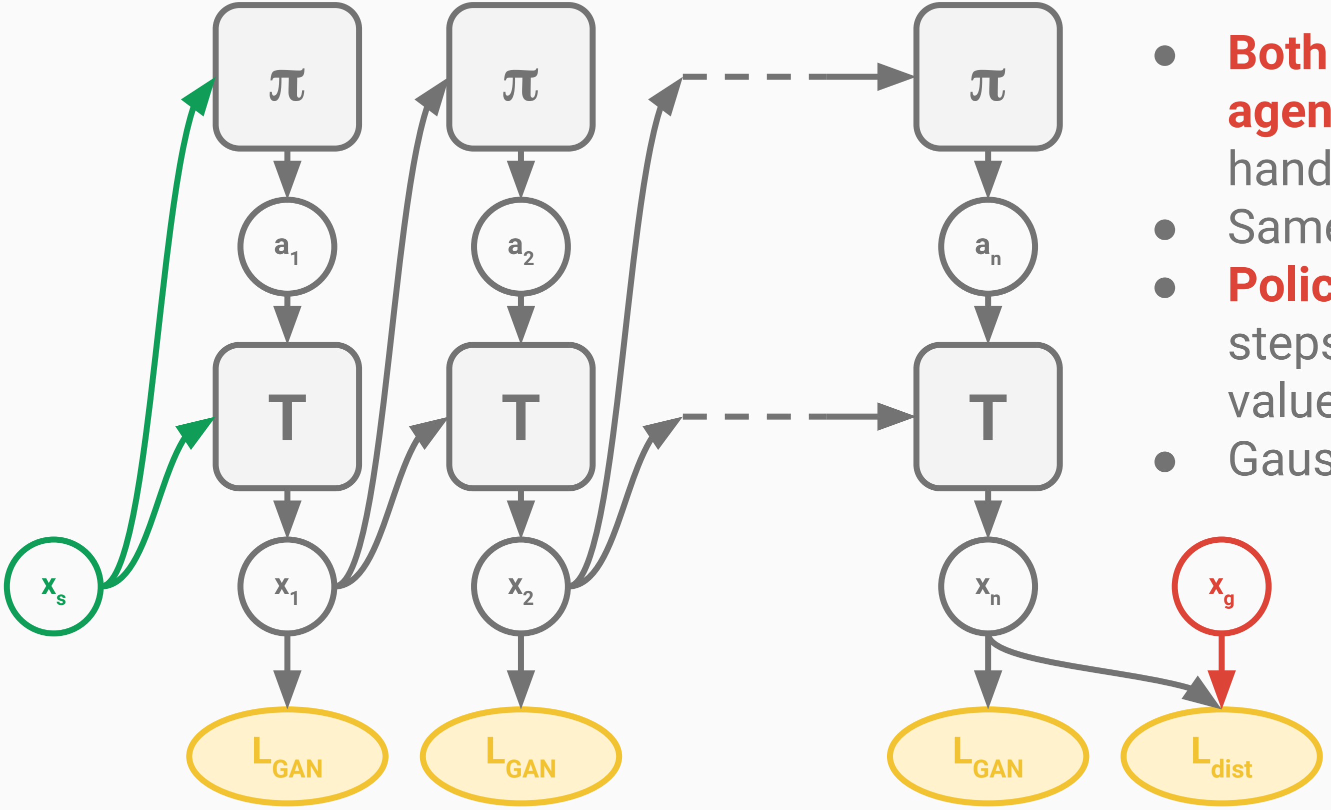
Model: Rewards

Four kinds of **reward** signals:

- Fixed negative reward for **each transition** in order to promote short trajectories
- Fixed negative reward if **maximum number of steps** exceeded
- **GAN**-based reward to make intermediate steps look like real images
- **Last step** is compared to $\mathbf{x}_g$ using some **distance** metric

Making all four work together is **tricky** so we focus **last two** (and fix number of steps)

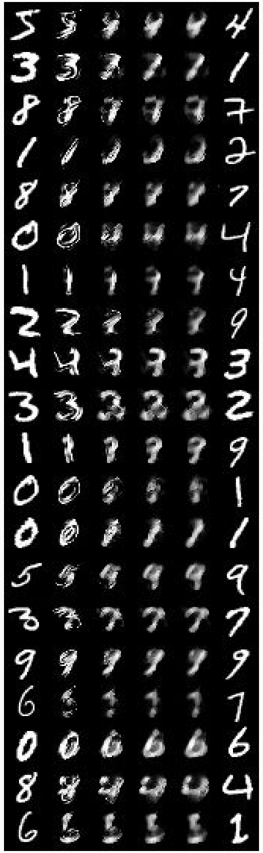
Model: Diagram and Training



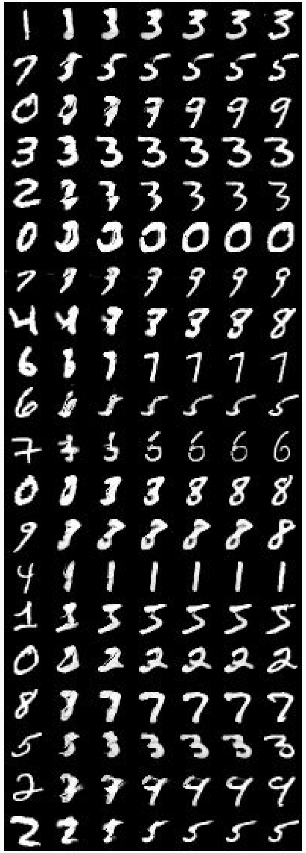
- **Both** π and **T** are treated as **agents** since it's hard to hand-design suitable **T**
- Same rewards
- **Policy gradients** (only 5 steps so no TD for the value estimate)
- Gaussian actions

Preliminary results (MNIST)

RL result:



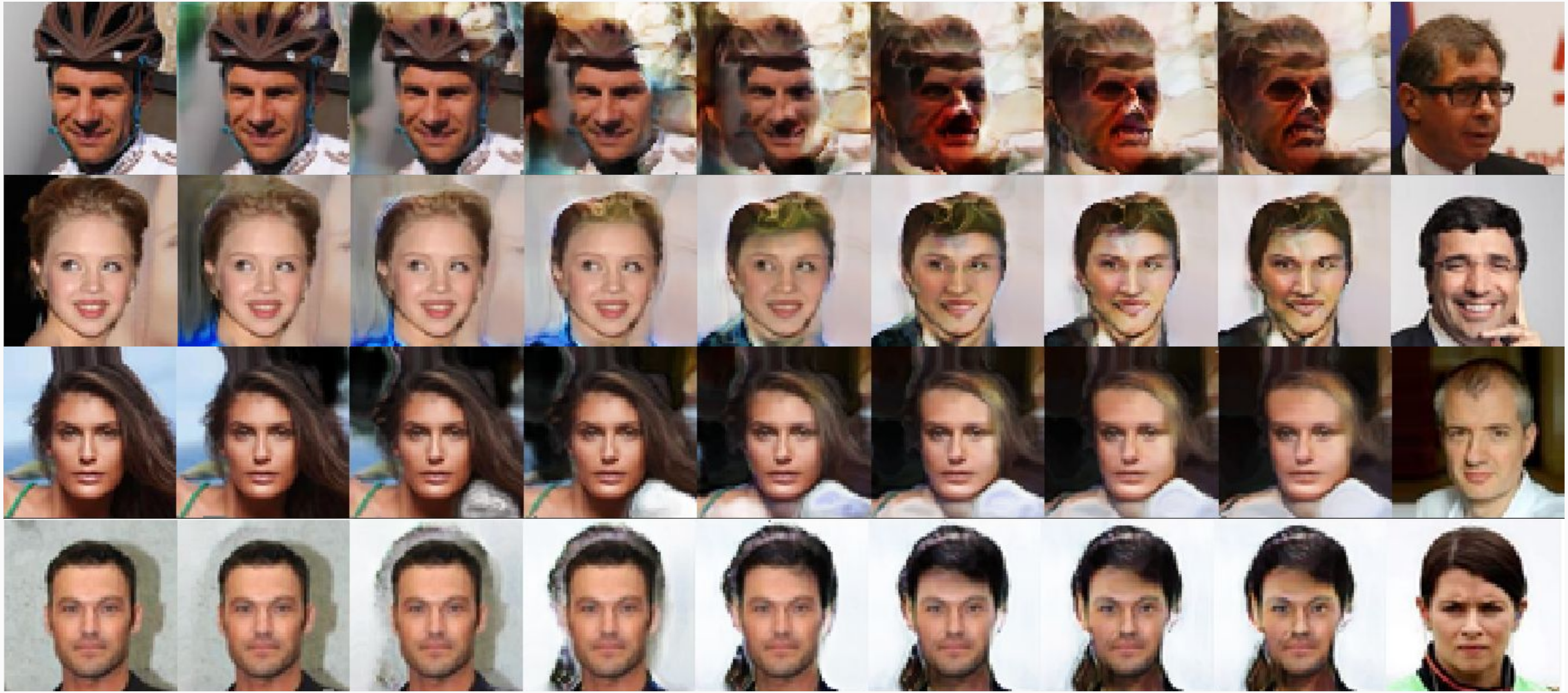
RNN result:



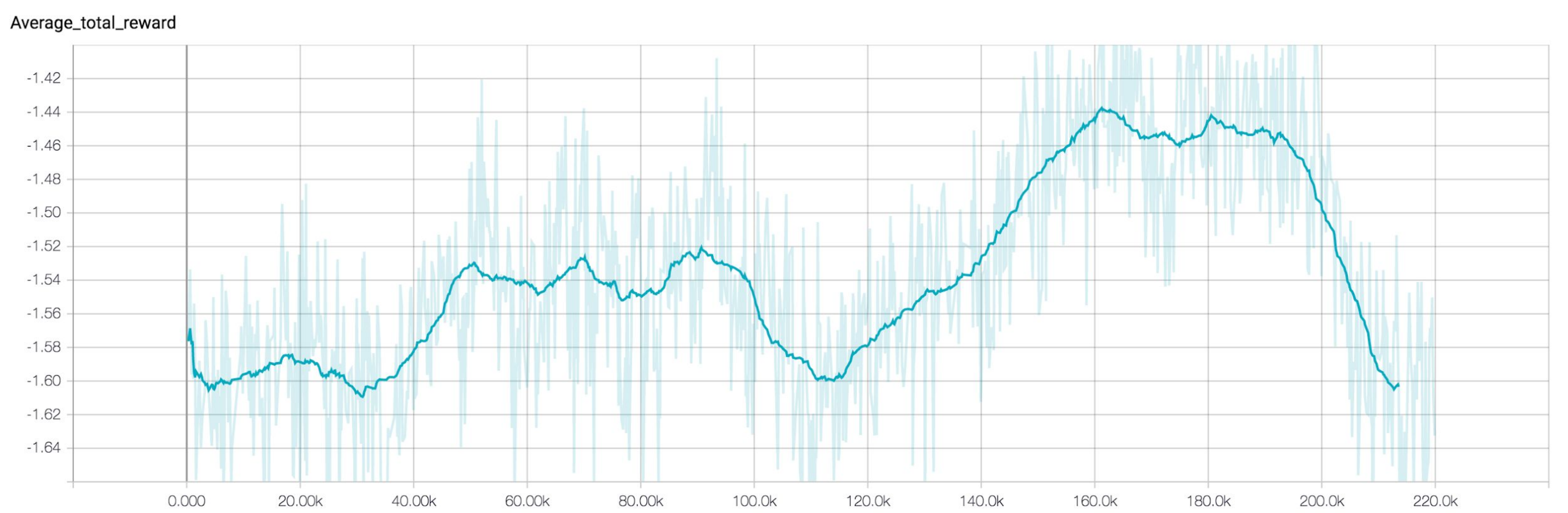
Observations:

- Much slower to train comparing to RNN
- Very unstable (tried a bunch of tricks like Huber loss for the value estimate but no luck)
- Does not require BPTT but TF can swap activations to save mem
- Couldn’t make RL/RNN-hybrid work (yet)

Fun stuff



Roller coaster


Average_total_reward
20.00k
40.00k
60.00k
80.00k
100.0k
120.0k
140.0k
160.0k
180.0k
200.0k
220.0k