

Breaking the cycle—Colleagues are all you need

Supplementary #4—Architecture

Ori Nizan
 Technion, Israel
 snizori@campus.technion.ac.il

Ayellet Tal
 Technion, Israel
 ayellet@ee.technion.ac.il

Generator-Encoder							
Layer	# filters	kernel_size	stride	padding	norm	activation	pad type
Conv2d	64	7	1	3	<i>in</i>	ReLU	<i>zero</i>
Conv2d	128	4	2	1	<i>in</i>	ReLU	<i>zero</i>
Conv2d	256	4	2	1	<i>in</i>	ReLU	<i>zero</i>
ResBlock 1							
Conv2d	256	3	1	1	<i>in</i>	ReLU	<i>zero</i>
Conv2d	256	3	1	1	<i>in</i>	None	<i>zero</i>
ResBlock 2							
Conv2d	256	3	1	1	<i>in</i>	ReLU	<i>zero</i>
Conv2d	256	3	1	1	<i>in</i>	None	<i>zero</i>
ResBlock 3							
Conv2d	256	3	1	1	<i>in</i>	ReLU	<i>zero</i>
Conv2d	256	3	1	1	<i>in</i>	None	<i>zero</i>
ResBlock 4							
Conv2d	256	3	1	1	<i>in</i>	ReLU	<i>zero</i>
Conv2d	256	3	1	1	<i>in</i>	None	<i>zero</i>
ResBlock 5							
Conv2d	256	3	1	1	<i>in</i>	ReLU	<i>zero</i>
Conv2d	256	3	1	1	<i>in</i>	None	<i>zero</i>

Figure 1: Encoder architecture.

Generator-Decoder							
Layer	# filters	kernel_size	stride	padding	norm	activation	pad type
ResBlock 1							
Conv2d	256	3	1	1	<i>adain</i>	ReLU	<i>zero</i>
Conv2d	256	3	1	1	<i>adain</i>	None	<i>zero</i>
ResBlock 2							
Conv2d	256	3	1	1	<i>adain</i>	ReLU	<i>zero</i>
Conv2d	256	3	1	1	<i>adain</i>	None	<i>zero</i>
ResBlock 3							
Conv2d	256	3	1	1	<i>adain</i>	ReLU	<i>zero</i>
Conv2d	256	3	1	1	<i>adain</i>	None	<i>zero</i>
ResBlock 4							
Conv2d	256	3	1	1	<i>adain</i>	ReLU	<i>zero</i>
Conv2d	256	3	1	1	<i>adain</i>	None	<i>zero</i>
ResBlock 5							
Conv2d	256	3	1	1	<i>adain</i>	ReLU	<i>zero</i>
Conv2d	256	3	1	1	<i>adain</i>	None	<i>zero</i>
Upsample	scale_factor=2				mode=nearest		
Conv2d	128	3	1	1	<i>adain</i>	ReLU	<i>zero</i>
Conv2d	128	3	1	1	<i>adain</i>	ReLU	<i>zero</i>
Upsample	scale_factor=2				mode=nearest		
Conv2d	128	3	1	1	<i>adain</i>	ReLU	<i>zero</i>
Conv2d	128	3	1	1	<i>adain</i>	ReLU	<i>zero</i>
Conv2d	64	1	1	0	<i>adain</i>	ReLU	<i>zero</i>
Conv2d	64	1	1	0	<i>adain</i>	ReLU	<i>zero</i>
Conv2d	# masks *4	1	1	0	None	<i>tanh</i>	<i>zero</i>
Mask	see caption						
MLP	see Table 2						

Table 1: **Decoder architecture.** This table describes the details of the generator’s decoder. Here, Mask = $\tanh(10 * netOutput[: -1 * \#masks]) \rightarrow$ last dimensions of the network output.

Layer	Input dimension	Output dimension	Normalization	Activation
LinearBlock	64	256	None	ReLU
LinearBlock	256	256	None	ReLU
LinearBlock	256	256	None	ReLU

Table 2: **MLP architecture.** This table shows the details of the last line in Table 2

Discriminator							
Layer	# filters	kernel_size	stride	padding	norm	activation	pad type
Conv2d	64	4	2	1	None	LReLU	<i>zero</i>
Conv2d	128	4	2	1	None	LReLU	<i>zero</i>
Conv2d	256	4	2	1	None	LReLU	<i>zero</i>
Conv2d	512	4	2	1	None	LReLU	<i>zero</i>
Conv2d	1	1	1	0			

Table 3: **Discriminator architecture.** Our discriminator is multi-scale, similarly to [21], with two scales.

Council Discriminator							
Layer	# filters	kernel_size	stride	padding	norm	activation	pad type
Conv2d	64	3	1	1	none	LReLU	zero
Conv2d	64	4	2	1	none	LReLU	zero
Conv2d	128	4	2	1	none	LReLU	zero
Conv2d	256	4	2	1	none	LReLU	zero
Conv2d	512	4	2	1	none	LReLU	zero
Conv2d	512	1	1	0			

Table 4: **Council discriminator architecture.** Recall that the input is a concatenation of the output and the input.