Guided diffusion with CLIP model

This network is a de-noising diffusion model and it is trained to reverse a gradual nosing process while allowing the model to generate samples images from the learned distributions starting from random noise. It is trained on continuous time-steps. The 'v' objective function is used from Progressive Distillation for Fast Sampling of Diffusion Models. CLIP guided diffusion mechanism is embedded into the current implementation. The diffusion model is conditioned on CLIP embeddings generated from input string prompts.

Note: A 602M parameter CLIP conditioned model trained for 3.1M steps and then fine-tuned for classifier-free guidance for 250K additional steps.

Code Usage:

Following parameters needs to be configured in **clip_sample.py** for generation of results. Default parameters are set but can be further changed to generate good quality results.

--prompts: The text string to be used for generation of images. Multiple input strings can be used by specifying relative weights for text prompts by putting the weight after a colon, for example: "the rise of consciousness:0.5".

--batch-size: Number of images needs to generated at same time(default 1)

--checkpoint: Path of the weights file to be used for diffusion model

--clip-guidance-scale: How strongly the result should match the text prompt (default 500). If set to 0, the cc12m_1 model will still be CLIP conditioned and sampling will go faster and use less memory.

--cutn: The number of random crops to compute CLIP embeddings for (default 16)

--cut-pow: The random crop size power (default 1)

--device: The PyTorch device name to use (default auto-detects)

--eta: Set to 0 (the default) while using --method ddim for deterministic (DDIM) sampling, 1 for stochastic (DDPM) sampling, and in between to interpolate between the two.

--images: The image prompts to use (local files or HTTP(S) URLs). Relative weights for image prompts can be specified by putting the weight after a colon, for example: "image_1.png:0.5".

--init: Specify the init image (optional: can be used in replacement of text promt)

--method: Specify the sampling method to use (DDPM, DDIM, PRK, PLMS, PIE, or PLMS2) (default DDPM). DDPM is the original SDE sampling method, DDIM integrates the probability flow ODE using a first order method, PLMS is fourth-order pseudo Adams-Bashforth, and PLMS2 is second-order pseudo Adams-Bashforth. PRK (fourth-order Pseudo Runge-Kutta) and PIE (second-order Pseudo Improved Euler) are used to bootstrap PLMS and PLMS2 but can be used on their own if you desire (slow).

--model: Specify the model to use (default cc12m_1)

-n: Number of samples(Images) from a single promt (default 1)

--seed: Specify the random seed (default 0)

--starting-timestep: Specify the starting time-step if an init image is used (range 0-1, default 0.9)

--size: Output image resolution (default auto)

--steps: specify the number of diffusion time-steps (default is 1000, can lower for faster but lower quality sampling)

Code Explained:

Code execution starts from setting generation parameters. Some parameters are for generation of resultant images and rest of parameters are for tuning the results. Generation of parameters involves (--prompts, --steps, --size, -n). Rest of the parameters can be left to default but tuning these parameters can generate better results.

The process of image generation starts from a random tensor and a continuous time-steps generated from input number of steps.

```
x = torch.randn([n, 3, side_y, side_x], device=device)
t = torch.linspace(1, 0, args.steps + 1, device=device)[:-1]
```

*Denoising Diffusion Probabilistic Model i*s used to integrate noise into time-steps for spliced DDPM/cosine noise schedule. Along with this a 512 dimensional embedding by parsing input prompt from CLIP model.

```
txt, weight = parse_prompt(prompt)
target_embeds.append(clip_model.encode_text(clip.tokenize(txt).to(device)).float())
weights.append(weight)
```

Random Noise vector along with time-steps and embedding from CLIP model are feed into diffusion network. A base image is generated through a simple straight forward pass and for better results generation image is passed through a sampling method.

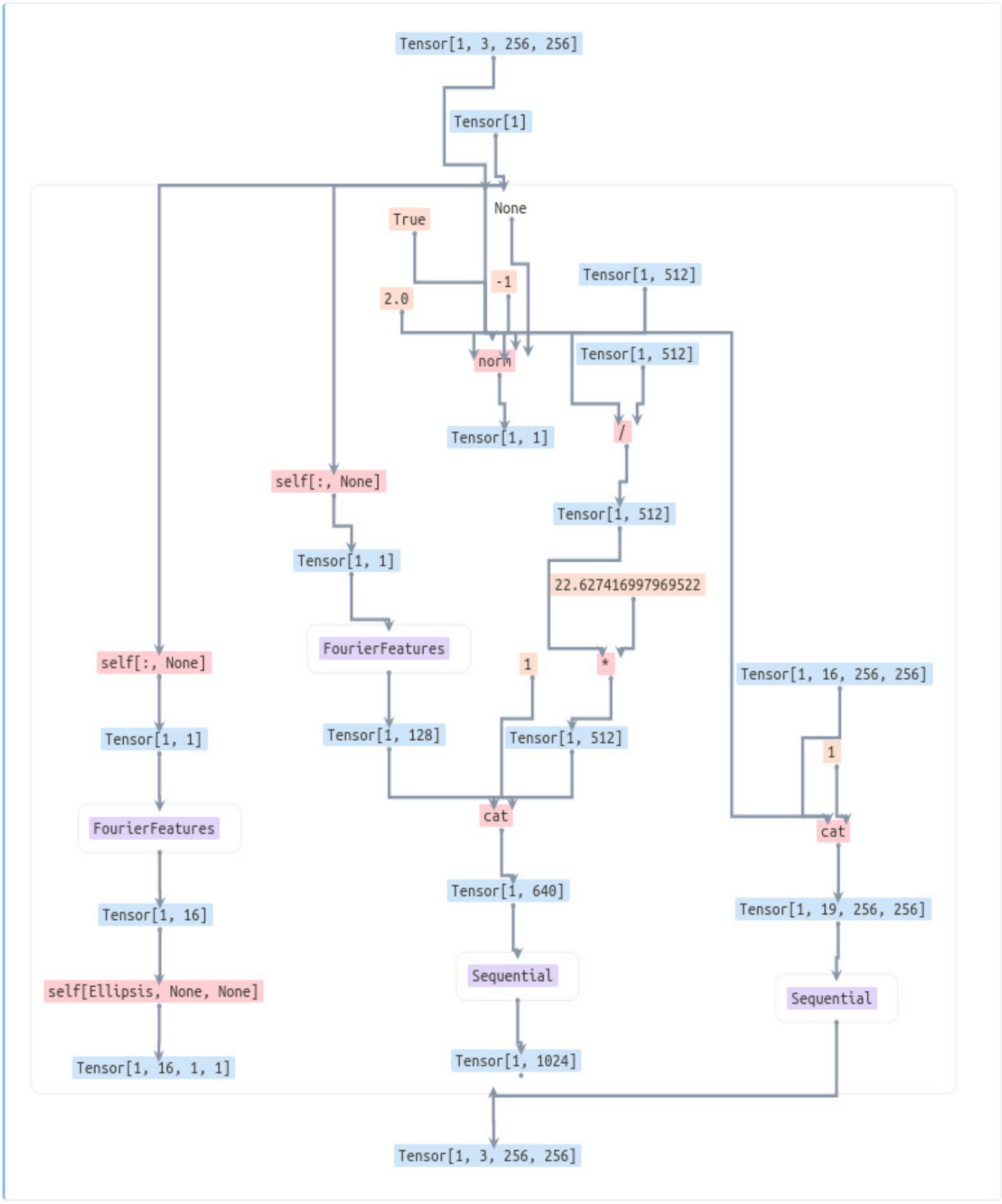
```
for i in trange(len(steps), disable=None):
    with torch.cuda.amp.autocast():
        v = model(x, ts * steps[i], **extra_args).float()
    pred = x * alphas[i] - v * sigmas[i]
    eps = x * sigmas[i] + v * alphas[i]
    if callback is not None:
        callback({'x': x, 'i': i, 't': steps[i], 'v': v, 'pred': pred})
    if i < len(steps) - 1:
        ddim_sigma = eta * (sigmas[i + 1]**2 / sigmas[i]**2).sqrt() * \
            (1 - alphas[i]**2 / alphas[i + 1]**2).sqrt()
        adjusted_sigma = (sigmas[i + 1]**2 - ddim_sigma**2).sqrt()
```

This process of sampling stay continuous for a defined number of steps and improves image quality step by stet.

Example Images:

Appendix:

```
(timestep_embed): FourierFeatures()
 (down): AvgPool2d(kernel_size=2, stride=2, padding=0)
 (up): Upsample(scale_factor=2.0, mode=bilinear)
 (net): Sequential(
  (0): ResModConvBlock(
   (main): Sequential(
    (0): Conv2d(19, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): GroupNorm(1, 128, eps=1e-05, affine=False)
    (2): Modulation2d(
     (layer): Linear(in_features=1024, out_features=256, bias=False)
    )
    (3): ReLU(inplace=True)
    (4): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 128, eps=1e-05, affine=False)
    (6): Modulation2d(
     (layer): Linear(in_features=1024, out_features=256, bias=False)
    )
    (7): ReLU(inplace=True)
   )
   (skip): Conv2d(19, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
  )
  (1): ResModConvBlock(
   (main): Sequential(
    (0): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): GroupNorm(1, 128, eps=1e-05, affine=False)
    (2): Modulation2d(
     (layer): Linear(in_features=1024, out_features=256, bias=False)
    )
    (3): ReLU(inplace=True)
    (4): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 128, eps=1e-05, affine=False)
    (6): Modulation2d(
     (layer): Linear(in_features=1024, out_features=256, bias=False)
    )
    (7): ReLU(inplace=True)
   )
   (skip): Identity()
  )
  (2): ResModConvBlock(
   (main): Sequential(
    (0): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): GroupNorm(1, 128, eps=1e-05, affine=False)
    (2): Modulation2d(
     (layer): Linear(in_features=1024, out_features=256, bias=False)
    )
    (3): ReLU(inplace=True)
    (4): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 128, eps=1e-05, affine=False)
    (6): Modulation2d(
     (layer): Linear(in_features=1024, out_features=256, bias=False)
    )
    (7): ReLU(inplace=True)
```



)
(skip): Identity()
)
(3): ResModConvBlock(

```
    (main): Sequential(
      (0): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
      (1): GroupNorm(1, 128, eps=1e-05, affine=False)
      (2): Modulation2d(
        (layer): Linear(in_features=1024, out_features=256, bias=False)
      )
      (3): ReLU(inplace=True)
      (4): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
      (5): GroupNorm(1, 128, eps=1e-05, affine=False)
      (6): Modulation2d(
        (layer): Linear(in_features=1024, out_features=256, bias=False)
      )
      (7): ReLU(inplace=True)
    )
    (skip): Identity()
  )
  (4): SkipBlock(
    (main): Sequential(
      (0): AvgPool2d(kernel_size=2, stride=2, padding=0)
      (1): ResModConvBlock(
        (main): Sequential(
          (0): Conv2d(128, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
          (1): GroupNorm(1, 256, eps=1e-05, affine=False)
          (2): Modulation2d(
            (layer): Linear(in_features=1024, out_features=512, bias=False)
          )
          (3): ReLU(inplace=True)
          (4): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
          (5): GroupNorm(1, 256, eps=1e-05, affine=False)
          (6): Modulation2d(
            (layer): Linear(in_features=1024, out_features=512, bias=False)
          )
          (7): ReLU(inplace=True)
        )
        (skip): Conv2d(128, 256, kernel_size=(1, 1), stride=(1, 1), bias=False)
      )
      (2): ResModConvBlock(
        (main): Sequential(
          (0): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
          (1): GroupNorm(1, 256, eps=1e-05, affine=False)
          (2): Modulation2d(
            (layer): Linear(in_features=1024, out_features=512, bias=False)
          )
          (3): ReLU(inplace=True)
          (4): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
          (5): GroupNorm(1, 256, eps=1e-05, affine=False)
          (6): Modulation2d(
            (layer): Linear(in_features=1024, out_features=512, bias=False)
          )
          (7): ReLU(inplace=True)
        )
        (skip): Identity()
      )
      (3): ResModConvBlock(
        (main): Sequential(
          (0): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
          (1): GroupNorm(1, 256, eps=1e-05, affine=False)
          (2): Modulation2d(
            (layer): Linear(in_features=1024, out_features=512, bias=False)
          )
          (3): ReLU(inplace=True)
          (4): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
          (5): GroupNorm(1, 256, eps=1e-05, affine=False)
          (6): Modulation2d(
            (layer): Linear(in_features=1024, out_features=512, bias=False)
          )
```

```
    (7): ReLU(inplace=True)
   )
  )
  (skip): Identity()
 )
 (4): ResModConvBlock(
  (main): Sequential(
   (0): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
   (1): GroupNorm(1, 256, eps=1e-05, affine=False)
   (2): Modulation2d(
    (layer): Linear(in_features=1024, out_features=512, bias=False)
   )
   (3): ReLU(inplace=True)
   (4): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
   (5): GroupNorm(1, 256, eps=1e-05, affine=False)
   (6): Modulation2d(
    (layer): Linear(in_features=1024, out_features=512, bias=False)
   )
   (7): ReLU(inplace=True)
  )
  (skip): Identity()
 )
 (5): SkipBlock(
  (main): Sequential(
   (0): AvgPool2d(kernel_size=2, stride=2, padding=0)
   (1): ResModConvBlock(
    (main): Sequential(
     (0): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
     (1): GroupNorm(1, 256, eps=1e-05, affine=False)
     (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=512, bias=False)
     )
     (3): ReLU(inplace=True)
     (4): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
     (5): GroupNorm(1, 256, eps=1e-05, affine=False)
     (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=512, bias=False)
     )
     (7): ReLU(inplace=True)
    )
    (skip): Identity()
   )
   (2): ResModConvBlock(
    (main): Sequential(
     (0): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
     (1): GroupNorm(1, 256, eps=1e-05, affine=False)
     (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=512, bias=False)
     )
     (3): ReLU(inplace=True)
     (4): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
     (5): GroupNorm(1, 256, eps=1e-05, affine=False)
     (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=512, bias=False)
     )
     (7): ReLU(inplace=True)
    )
    (skip): Identity()
   )
   (3): ResModConvBlock(
    (main): Sequential(
     (0): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
     (1): GroupNorm(1, 256, eps=1e-05, affine=False)
     (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=512, bias=False)
     )
     (3): ReLU(inplace=True)
```

```
    (4): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 256, eps=1e-05, affine=False)
    (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=512, bias=False)
    )
    (7): ReLU(inplace=True)
  )
  (skip): Identity()
)
(4): ResModConvBlock(
  (main): Sequential(
    (0): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): GroupNorm(1, 256, eps=1e-05, affine=False)
    (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=512, bias=False)
    )
    (3): ReLU(inplace=True)
    (4): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 256, eps=1e-05, affine=False)
    (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=512, bias=False)
    )
    (7): ReLU(inplace=True)
  )
  (skip): Identity()
)
(5): SkipBlock(
  (main): Sequential(
    (0): AvgPool2d(kernel_size=2, stride=2, padding=0)
    (1): ResModConvBlock(
      (main): Sequential(
        (0): Conv2d(256, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
        (1): GroupNorm(1, 512, eps=1e-05, affine=False)
        (2): Modulation2d(
          (layer): Linear(in_features=1024, out_features=1024, bias=False)
        )
        (3): ReLU(inplace=True)
        (4): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
        (5): GroupNorm(1, 512, eps=1e-05, affine=False)
        (6): Modulation2d(
          (layer): Linear(in_features=1024, out_features=1024, bias=False)
        )
        (7): ReLU(inplace=True)
      )
      (skip): Conv2d(256, 512, kernel_size=(1, 1), stride=(1, 1), bias=False)
    )
    (2): ResModConvBlock(
      (main): Sequential(
        (0): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
        (1): GroupNorm(1, 512, eps=1e-05, affine=False)
        (2): Modulation2d(
          (layer): Linear(in_features=1024, out_features=1024, bias=False)
        )
        (3): ReLU(inplace=True)
        (4): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
        (5): GroupNorm(1, 512, eps=1e-05, affine=False)
        (6): Modulation2d(
          (layer): Linear(in_features=1024, out_features=1024, bias=False)
        )
        (7): ReLU(inplace=True)
      )
      (skip): Identity()
    )
    (3): ResModConvBlock(
      (main): Sequential(
        (0): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
```

```
    (1): GroupNorm(1, 512, eps=1e-05, affine=False)
    (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=1024, bias=False)
    )
    (3): ReLU(inplace=True)
    (4): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 512, eps=1e-05, affine=False)
    (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=1024, bias=False)
    )
    (7): ReLU(inplace=True)
  )
  (skip): Identity()
)
(4): ResModConvBlock(
  (main): Sequential(
    (0): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): GroupNorm(1, 512, eps=1e-05, affine=False)
    (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=1024, bias=False)
    )
    (3): ReLU(inplace=True)
    (4): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 512, eps=1e-05, affine=False)
    (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=1024, bias=False)
    )
    (7): ReLU(inplace=True)
  )
  (skip): Identity()
)
(5): SkipBlock(
  (main): Sequential(
    (0): AvgPool2d(kernel_size=2, stride=2, padding=0)
    (1): ResModConvBlock(
      (main): Sequential(
        (0): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
        (1): GroupNorm(1, 512, eps=1e-05, affine=False)
        (2): Modulation2d(
          (layer): Linear(in_features=1024, out_features=1024, bias=False)
        )
        (3): ReLU(inplace=True)
        (4): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
        (5): GroupNorm(1, 512, eps=1e-05, affine=False)
        (6): Modulation2d(
          (layer): Linear(in_features=1024, out_features=1024, bias=False)
        )
        (7): ReLU(inplace=True)
      )
      (skip): Identity()
    )
    (2): SelfAttention2d(
      (norm): GroupNorm(1, 512, eps=1e-05, affine=True)
      (qkv_proj): Conv2d(512, 1536, kernel_size=(1, 1), stride=(1, 1))
      (out_proj): Conv2d(512, 512, kernel_size=(1, 1), stride=(1, 1))
      (dropout): Identity()
    )
    (3): ResModConvBlock(
      (main): Sequential(
        (0): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
        (1): GroupNorm(1, 512, eps=1e-05, affine=False)
        (2): Modulation2d(
          (layer): Linear(in_features=1024, out_features=1024, bias=False)
        )
        (3): ReLU(inplace=True)
        (4): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
```

```
    (5): GroupNorm(1, 512, eps=1e-05, affine=False)
    (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=1024, bias=False)
    )
    (7): ReLU(inplace=True)
  )
  (skip): Identity()
)
(4): SelfAttention2d(
  (norm): GroupNorm(1, 512, eps=1e-05, affine=True)
  (qkv_proj): Conv2d(512, 1536, kernel_size=(1, 1), stride=(1, 1))
  (out_proj): Conv2d(512, 512, kernel_size=(1, 1), stride=(1, 1))
  (dropout): Identity()
)
(5): ResModConvBlock(
  (main): Sequential(
    (0): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): GroupNorm(1, 512, eps=1e-05, affine=False)
    (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=1024, bias=False)
    )
    (3): ReLU(inplace=True)
    (4): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 512, eps=1e-05, affine=False)
    (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=1024, bias=False)
    )
    (7): ReLU(inplace=True)
  )
  (skip): Identity()
)
(6): SelfAttention2d(
  (norm): GroupNorm(1, 512, eps=1e-05, affine=True)
  (qkv_proj): Conv2d(512, 1536, kernel_size=(1, 1), stride=(1, 1))
  (out_proj): Conv2d(512, 512, kernel_size=(1, 1), stride=(1, 1))
  (dropout): Identity()
)
(7): ResModConvBlock(
  (main): Sequential(
    (0): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): GroupNorm(1, 512, eps=1e-05, affine=False)
    (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=1024, bias=False)
    )
    (3): ReLU(inplace=True)
    (4): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 512, eps=1e-05, affine=False)
    (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=1024, bias=False)
    )
    (7): ReLU(inplace=True)
  )
  (skip): Identity()
)
(8): SelfAttention2d(
  (norm): GroupNorm(1, 512, eps=1e-05, affine=True)
  (qkv_proj): Conv2d(512, 1536, kernel_size=(1, 1), stride=(1, 1))
  (out_proj): Conv2d(512, 512, kernel_size=(1, 1), stride=(1, 1))
  (dropout): Identity()
)
(9): SkipBlock(
  (main): Sequential(
    (0): AvgPool2d(kernel_size=2, stride=2, padding=0)
    (1): ResModConvBlock(
      (main): Sequential(
        (0): Conv2d(512, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
```

```
    (1): GroupNorm(1, 1024, eps=1e-05, affine=False)
    (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=2048, bias=False)
    )
    (3): ReLU(inplace=True)
    (4): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 1024, eps=1e-05, affine=False)
    (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=2048, bias=False)
    )
    (7): ReLU(inplace=True)
  )
  (skip): Conv2d(512, 1024, kernel_size=(1, 1), stride=(1, 1), bias=False)
)
(2): SelfAttention2d(
  (norm): GroupNorm(1, 1024, eps=1e-05, affine=True)
  (qkv_proj): Conv2d(1024, 3072, kernel_size=(1, 1), stride=(1, 1))
  (out_proj): Conv2d(1024, 1024, kernel_size=(1, 1), stride=(1, 1))
  (dropout): Identity()
)
(3): ResModConvBlock(
  (main): Sequential(
    (0): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): GroupNorm(1, 1024, eps=1e-05, affine=False)
    (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=2048, bias=False)
    )
    (3): ReLU(inplace=True)
    (4): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 1024, eps=1e-05, affine=False)
    (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=2048, bias=False)
    )
    (7): ReLU(inplace=True)
  )
  (skip): Identity()
)
(4): SelfAttention2d(
  (norm): GroupNorm(1, 1024, eps=1e-05, affine=True)
  (qkv_proj): Conv2d(1024, 3072, kernel_size=(1, 1), stride=(1, 1))
  (out_proj): Conv2d(1024, 1024, kernel_size=(1, 1), stride=(1, 1))
  (dropout): Identity()
)
(5): ResModConvBlock(
  (main): Sequential(
    (0): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): GroupNorm(1, 1024, eps=1e-05, affine=False)
    (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=2048, bias=False)
    )
    (3): ReLU(inplace=True)
    (4): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 1024, eps=1e-05, affine=False)
    (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=2048, bias=False)
    )
    (7): ReLU(inplace=True)
  )
  (skip): Identity()
)
(6): SelfAttention2d(
  (norm): GroupNorm(1, 1024, eps=1e-05, affine=True)
  (qkv_proj): Conv2d(1024, 3072, kernel_size=(1, 1), stride=(1, 1))
  (out_proj): Conv2d(1024, 1024, kernel_size=(1, 1), stride=(1, 1))
  (dropout): Identity()
)
```

```
(7): ResModConvBlock(
  (main): Sequential(
    (0): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): GroupNorm(1, 1024, eps=1e-05, affine=False)
    (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=2048, bias=False)
    )
    (3): ReLU(inplace=True)
    (4): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 1024, eps=1e-05, affine=False)
    (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=2048, bias=False)
    )
    (7): ReLU(inplace=True)
  )
  (skip): Identity()
)
(8): SelfAttention2d(
  (norm): GroupNorm(1, 1024, eps=1e-05, affine=True)
  (qkv_proj): Conv2d(1024, 3072, kernel_size=(1, 1), stride=(1, 1))
  (out_proj): Conv2d(1024, 1024, kernel_size=(1, 1), stride=(1, 1))
  (dropout): Identity()
)
(9): SkipBlock(
  (main): Sequential(
    (0): AvgPool2d(kernel_size=2, stride=2, padding=0)
    (1): ResModConvBlock(
      (main): Sequential(
        (0): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
        (1): GroupNorm(1, 1024, eps=1e-05, affine=False)
        (2): Modulation2d(
          (layer): Linear(in_features=1024, out_features=2048, bias=False)
        )
        (3): ReLU(inplace=True)
        (4): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
        (5): GroupNorm(1, 1024, eps=1e-05, affine=False)
        (6): Modulation2d(
          (layer): Linear(in_features=1024, out_features=2048, bias=False)
        )
        (7): ReLU(inplace=True)
      )
      (skip): Identity()
    )
    (2): SelfAttention2d(
      (norm): GroupNorm(1, 1024, eps=1e-05, affine=True)
      (qkv_proj): Conv2d(1024, 3072, kernel_size=(1, 1), stride=(1, 1))
      (out_proj): Conv2d(1024, 1024, kernel_size=(1, 1), stride=(1, 1))
      (dropout): Identity()
    )
    (3): ResModConvBlock(
      (main): Sequential(
        (0): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
        (1): GroupNorm(1, 1024, eps=1e-05, affine=False)
        (2): Modulation2d(
          (layer): Linear(in_features=1024, out_features=2048, bias=False)
        )
        (3): ReLU(inplace=True)
        (4): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
        (5): GroupNorm(1, 1024, eps=1e-05, affine=False)
        (6): Modulation2d(
          (layer): Linear(in_features=1024, out_features=2048, bias=False)
        )
        (7): ReLU(inplace=True)
      )
      (skip): Identity()
    )
```

```
(4): SelfAttention2d(
  (norm): GroupNorm(1, 1024, eps=1e-05, affine=True)
  (qkv_proj): Conv2d(1024, 3072, kernel_size=(1, 1), stride=(1, 1))
  (out_proj): Conv2d(1024, 1024, kernel_size=(1, 1), stride=(1, 1))
  (dropout): Identity()
)
(5): ResModConvBlock(
  (main): Sequential(
    (0): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): GroupNorm(1, 1024, eps=1e-05, affine=False)
    (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=2048, bias=False)
    )
    (3): ReLU(inplace=True)
    (4): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 1024, eps=1e-05, affine=False)
    (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=2048, bias=False)
    )
    (7): ReLU(inplace=True)
  )
  (skip): Identity()
)
(6): SelfAttention2d(
  (norm): GroupNorm(1, 1024, eps=1e-05, affine=True)
  (qkv_proj): Conv2d(1024, 3072, kernel_size=(1, 1), stride=(1, 1))
  (out_proj): Conv2d(1024, 1024, kernel_size=(1, 1), stride=(1, 1))
  (dropout): Identity()
)
(7): ResModConvBlock(
  (main): Sequential(
    (0): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): GroupNorm(1, 1024, eps=1e-05, affine=False)
    (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=2048, bias=False)
    )
    (3): ReLU(inplace=True)
    (4): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 1024, eps=1e-05, affine=False)
    (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=2048, bias=False)
    )
    (7): ReLU(inplace=True)
  )
  (skip): Identity()
)
(8): SelfAttention2d(
  (norm): GroupNorm(1, 1024, eps=1e-05, affine=True)
  (qkv_proj): Conv2d(1024, 3072, kernel_size=(1, 1), stride=(1, 1))
  (out_proj): Conv2d(1024, 1024, kernel_size=(1, 1), stride=(1, 1))
  (dropout): Identity()
)
(9): ResModConvBlock(
  (main): Sequential(
    (0): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): GroupNorm(1, 1024, eps=1e-05, affine=False)
    (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=2048, bias=False)
    )
    (3): ReLU(inplace=True)
    (4): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 1024, eps=1e-05, affine=False)
    (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=2048, bias=False)
    )
    (7): ReLU(inplace=True)
```

```
  )
  (skip): Identity()
)
(10): SelfAttention2d(
  (norm): GroupNorm(1, 1024, eps=1e-05, affine=True)
  (qkv_proj): Conv2d(1024, 3072, kernel_size=(1, 1), stride=(1, 1))
  (out_proj): Conv2d(1024, 1024, kernel_size=(1, 1), stride=(1, 1))
  (dropout): Identity()
)
(11): ResModConvBlock(
  (main): Sequential(
    (0): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): GroupNorm(1, 1024, eps=1e-05, affine=False)
    (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=2048, bias=False)
    )
    (3): ReLU(inplace=True)
    (4): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 1024, eps=1e-05, affine=False)
    (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=2048, bias=False)
    )
    (7): ReLU(inplace=True)
  )
  (skip): Identity()
)
(12): SelfAttention2d(
  (norm): GroupNorm(1, 1024, eps=1e-05, affine=True)
  (qkv_proj): Conv2d(1024, 3072, kernel_size=(1, 1), stride=(1, 1))
  (out_proj): Conv2d(1024, 1024, kernel_size=(1, 1), stride=(1, 1))
  (dropout): Identity()
)
(13): ResModConvBlock(
  (main): Sequential(
    (0): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): GroupNorm(1, 1024, eps=1e-05, affine=False)
    (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=2048, bias=False)
    )
    (3): ReLU(inplace=True)
    (4): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 1024, eps=1e-05, affine=False)
    (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=2048, bias=False)
    )
    (7): ReLU(inplace=True)
  )
  (skip): Identity()
)
(14): SelfAttention2d(
  (norm): GroupNorm(1, 1024, eps=1e-05, affine=True)
  (qkv_proj): Conv2d(1024, 3072, kernel_size=(1, 1), stride=(1, 1))
  (out_proj): Conv2d(1024, 1024, kernel_size=(1, 1), stride=(1, 1))
  (dropout): Identity()
)
(15): ResModConvBlock(
  (main): Sequential(
    (0): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): GroupNorm(1, 1024, eps=1e-05, affine=False)
    (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=2048, bias=False)
    )
    (3): ReLU(inplace=True)
    (4): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 1024, eps=1e-05, affine=False)
    (6): Modulation2d(
```

```
        (layer): Linear(in_features=1024, out_features=2048, bias=False)
       )
       (7): ReLU(inplace=True)
      )
      (skip): Identity()
     )
     (16): SelfAttention2d(
      (norm): GroupNorm(1, 1024, eps=1e-05, affine=True)
      (qkv_proj): Conv2d(1024, 3072, kernel_size=(1, 1), stride=(1, 1))
      (out_proj): Conv2d(1024, 1024, kernel_size=(1, 1), stride=(1, 1))
      (dropout): Identity()
     )
     (17): Upsample(scale_factor=2.0, mode=bilinear)
    )
    (skip): Identity()
   )
   (10): ResModConvBlock(
    (main): Sequential(
     (0): Conv2d(2048, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
     (1): GroupNorm(1, 1024, eps=1e-05, affine=False)
     (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=2048, bias=False)
     )
     (3): ReLU(inplace=True)
     (4): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
     (5): GroupNorm(1, 1024, eps=1e-05, affine=False)
     (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=2048, bias=False)
     )
     (7): ReLU(inplace=True)
    )
    (skip): Conv2d(2048, 1024, kernel_size=(1, 1), stride=(1, 1), bias=False)
   )
   (11): SelfAttention2d(
    (norm): GroupNorm(1, 1024, eps=1e-05, affine=True)
    (qkv_proj): Conv2d(1024, 3072, kernel_size=(1, 1), stride=(1, 1))
    (out_proj): Conv2d(1024, 1024, kernel_size=(1, 1), stride=(1, 1))
    (dropout): Identity()
   )
   (12): ResModConvBlock(
    (main): Sequential(
     (0): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
     (1): GroupNorm(1, 1024, eps=1e-05, affine=False)
     (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=2048, bias=False)
     )
     (3): ReLU(inplace=True)
     (4): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
     (5): GroupNorm(1, 1024, eps=1e-05, affine=False)
     (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=2048, bias=False)
     )
     (7): ReLU(inplace=True)
    )
    (skip): Identity()
   )
   (13): SelfAttention2d(
    (norm): GroupNorm(1, 1024, eps=1e-05, affine=True)
    (qkv_proj): Conv2d(1024, 3072, kernel_size=(1, 1), stride=(1, 1))
    (out_proj): Conv2d(1024, 1024, kernel_size=(1, 1), stride=(1, 1))
    (dropout): Identity()
   )
   (14): ResModConvBlock(
    (main): Sequential(
     (0): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
     (1): GroupNorm(1, 1024, eps=1e-05, affine=False)
```

```
      (2): Modulation2d(
        (layer): Linear(in_features=1024, out_features=2048, bias=False)
      )
      (3): ReLU(inplace=True)
      (4): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
      (5): GroupNorm(1, 1024, eps=1e-05, affine=False)
      (6): Modulation2d(
        (layer): Linear(in_features=1024, out_features=2048, bias=False)
      )
      (7): ReLU(inplace=True)
    )
    (skip): Identity()
  )
  (15): SelfAttention2d(
    (norm): GroupNorm(1, 1024, eps=1e-05, affine=True)
    (qkv_proj): Conv2d(1024, 3072, kernel_size=(1, 1), stride=(1, 1))
    (out_proj): Conv2d(1024, 1024, kernel_size=(1, 1), stride=(1, 1))
    (dropout): Identity()
  )
  (16): ResModConvBlock(
    (main): Sequential(
      (0): Conv2d(1024, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
      (1): GroupNorm(1, 1024, eps=1e-05, affine=False)
      (2): Modulation2d(
        (layer): Linear(in_features=1024, out_features=2048, bias=False)
      )
      (3): ReLU(inplace=True)
      (4): Conv2d(1024, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
      (5): GroupNorm(1, 512, eps=1e-05, affine=False)
      (6): Modulation2d(
        (layer): Linear(in_features=1024, out_features=1024, bias=False)
      )
      (7): ReLU(inplace=True)
    )
    (skip): Conv2d(1024, 512, kernel_size=(1, 1), stride=(1, 1), bias=False)
  )
  (17): SelfAttention2d(
    (norm): GroupNorm(1, 512, eps=1e-05, affine=True)
    (qkv_proj): Conv2d(512, 1536, kernel_size=(1, 1), stride=(1, 1))
    (out_proj): Conv2d(512, 512, kernel_size=(1, 1), stride=(1, 1))
    (dropout): Identity()
  )
  (18): Upsample(scale_factor=2.0, mode=bilinear)
 )
 (skip): Identity()
)
(10): ResModConvBlock(
 (main): Sequential(
  (0): Conv2d(1024, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (1): GroupNorm(1, 512, eps=1e-05, affine=False)
  (2): Modulation2d(
    (layer): Linear(in_features=1024, out_features=1024, bias=False)
  )
  (3): ReLU(inplace=True)
  (4): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (5): GroupNorm(1, 512, eps=1e-05, affine=False)
  (6): Modulation2d(
    (layer): Linear(in_features=1024, out_features=1024, bias=False)
  )
  (7): ReLU(inplace=True)
 )
 (skip): Conv2d(1024, 512, kernel_size=(1, 1), stride=(1, 1), bias=False)
)
(11): SelfAttention2d(
 (norm): GroupNorm(1, 512, eps=1e-05, affine=True)
 (qkv_proj): Conv2d(512, 1536, kernel_size=(1, 1), stride=(1, 1))
```

```
  (out_proj): Conv2d(512, 512, kernel_size=(1, 1), stride=(1, 1))
  (dropout): Identity()
 )
 (12): ResModConvBlock(
  (main): Sequential(
   (0): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
   (1): GroupNorm(1, 512, eps=1e-05, affine=False)
   (2): Modulation2d(
    (layer): Linear(in_features=1024, out_features=1024, bias=False)
   )
   (3): ReLU(inplace=True)
   (4): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
   (5): GroupNorm(1, 512, eps=1e-05, affine=False)
   (6): Modulation2d(
    (layer): Linear(in_features=1024, out_features=1024, bias=False)
   )
   (7): ReLU(inplace=True)
  )
  (skip): Identity()
 )
 (13): SelfAttention2d(
  (norm): GroupNorm(1, 512, eps=1e-05, affine=True)
  (qkv_proj): Conv2d(512, 1536, kernel_size=(1, 1), stride=(1, 1))
  (out_proj): Conv2d(512, 512, kernel_size=(1, 1), stride=(1, 1))
  (dropout): Identity()
 )
 (14): ResModConvBlock(
  (main): Sequential(
   (0): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
   (1): GroupNorm(1, 512, eps=1e-05, affine=False)
   (2): Modulation2d(
    (layer): Linear(in_features=1024, out_features=1024, bias=False)
   )
   (3): ReLU(inplace=True)
   (4): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
   (5): GroupNorm(1, 512, eps=1e-05, affine=False)
   (6): Modulation2d(
    (layer): Linear(in_features=1024, out_features=1024, bias=False)
   )
   (7): ReLU(inplace=True)
  )
  (skip): Identity()
 )
 (15): SelfAttention2d(
  (norm): GroupNorm(1, 512, eps=1e-05, affine=True)
  (qkv_proj): Conv2d(512, 1536, kernel_size=(1, 1), stride=(1, 1))
  (out_proj): Conv2d(512, 512, kernel_size=(1, 1), stride=(1, 1))
  (dropout): Identity()
 )
 (16): ResModConvBlock(
  (main): Sequential(
   (0): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
   (1): GroupNorm(1, 512, eps=1e-05, affine=False)
   (2): Modulation2d(
    (layer): Linear(in_features=1024, out_features=1024, bias=False)
   )
   (3): ReLU(inplace=True)
   (4): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
   (5): GroupNorm(1, 512, eps=1e-05, affine=False)
   (6): Modulation2d(
    (layer): Linear(in_features=1024, out_features=1024, bias=False)
   )
   (7): ReLU(inplace=True)
  )
  (skip): Identity()
 )
```

```
    (17): SelfAttention2d(
      (norm): GroupNorm(1, 512, eps=1e-05, affine=True)
      (qkv_proj): Conv2d(512, 1536, kernel_size=(1, 1), stride=(1, 1))
      (out_proj): Conv2d(512, 512, kernel_size=(1, 1), stride=(1, 1))
      (dropout): Identity()
    )
    (18): Upsample(scale_factor=2.0, mode=bilinear)
  )
  (skip): Identity()
)
(6): ResModConvBlock(
  (main): Sequential(
    (0): Conv2d(1024, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): GroupNorm(1, 512, eps=1e-05, affine=False)
    (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=1024, bias=False)
    )
    (3): ReLU(inplace=True)
    (4): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 512, eps=1e-05, affine=False)
    (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=1024, bias=False)
    )
    (7): ReLU(inplace=True)
  )
  (skip): Conv2d(1024, 512, kernel_size=(1, 1), stride=(1, 1), bias=False)
)
(7): ResModConvBlock(
  (main): Sequential(
    (0): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): GroupNorm(1, 512, eps=1e-05, affine=False)
    (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=1024, bias=False)
    )
    (3): ReLU(inplace=True)
    (4): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 512, eps=1e-05, affine=False)
    (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=1024, bias=False)
    )
    (7): ReLU(inplace=True)
  )
  (skip): Identity()
)
(8): ResModConvBlock(
  (main): Sequential(
    (0): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): GroupNorm(1, 512, eps=1e-05, affine=False)
    (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=1024, bias=False)
    )
    (3): ReLU(inplace=True)
    (4): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 512, eps=1e-05, affine=False)
    (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=1024, bias=False)
    )
    (7): ReLU(inplace=True)
  )
  (skip): Identity()
)
(9): ResModConvBlock(
  (main): Sequential(
    (0): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): GroupNorm(1, 512, eps=1e-05, affine=False)
    (2): Modulation2d(
```

```
        (layer): Linear(in_features=1024, out_features=1024, bias=False)
      )
      (3): ReLU(inplace=True)
      (4): Conv2d(512, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
      (5): GroupNorm(1, 256, eps=1e-05, affine=False)
      (6): Modulation2d(
        (layer): Linear(in_features=1024, out_features=512, bias=False)
      )
      (7): ReLU(inplace=True)
    )
    (skip): Conv2d(512, 256, kernel_size=(1, 1), stride=(1, 1), bias=False)
  )
  (10): Upsample(scale_factor=2.0, mode=bilinear)
 )
 (skip): Identity()
)
(6): ResModConvBlock(
 (main): Sequential(
  (0): Conv2d(512, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (1): GroupNorm(1, 256, eps=1e-05, affine=False)
  (2): Modulation2d(
    (layer): Linear(in_features=1024, out_features=512, bias=False)
  )
  (3): ReLU(inplace=True)
  (4): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (5): GroupNorm(1, 256, eps=1e-05, affine=False)
  (6): Modulation2d(
    (layer): Linear(in_features=1024, out_features=512, bias=False)
  )
  (7): ReLU(inplace=True)
 )
 (skip): Conv2d(512, 256, kernel_size=(1, 1), stride=(1, 1), bias=False)
)
(7): ResModConvBlock(
 (main): Sequential(
  (0): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (1): GroupNorm(1, 256, eps=1e-05, affine=False)
  (2): Modulation2d(
    (layer): Linear(in_features=1024, out_features=512, bias=False)
  )
  (3): ReLU(inplace=True)
  (4): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (5): GroupNorm(1, 256, eps=1e-05, affine=False)
  (6): Modulation2d(
    (layer): Linear(in_features=1024, out_features=512, bias=False)
  )
  (7): ReLU(inplace=True)
 )
 (skip): Identity()
)
(8): ResModConvBlock(
 (main): Sequential(
  (0): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (1): GroupNorm(1, 256, eps=1e-05, affine=False)
  (2): Modulation2d(
    (layer): Linear(in_features=1024, out_features=512, bias=False)
  )
  (3): ReLU(inplace=True)
  (4): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (5): GroupNorm(1, 256, eps=1e-05, affine=False)
  (6): Modulation2d(
    (layer): Linear(in_features=1024, out_features=512, bias=False)
  )
  (7): ReLU(inplace=True)
 )
 (skip): Identity()
```

```
    )
    (9): ResModConvBlock(
      (main): Sequential(
        (0): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
        (1): GroupNorm(1, 256, eps=1e-05, affine=False)
        (2): Modulation2d(
          (layer): Linear(in_features=1024, out_features=512, bias=False)
        )
        (3): ReLU(inplace=True)
        (4): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
        (5): GroupNorm(1, 256, eps=1e-05, affine=False)
        (6): Modulation2d(
          (layer): Linear(in_features=1024, out_features=512, bias=False)
        )
        (7): ReLU(inplace=True)
      )
      (skip): Identity()
    )
    (10): Upsample(scale_factor=2.0, mode=bilinear)
  )
  (skip): Identity()
)
(6): ResModConvBlock(
  (main): Sequential(
    (0): Conv2d(512, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): GroupNorm(1, 256, eps=1e-05, affine=False)
    (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=512, bias=False)
    )
    (3): ReLU(inplace=True)
    (4): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 256, eps=1e-05, affine=False)
    (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=512, bias=False)
    )
    (7): ReLU(inplace=True)
  )
  (skip): Conv2d(512, 256, kernel_size=(1, 1), stride=(1, 1), bias=False)
)
(7): ResModConvBlock(
  (main): Sequential(
    (0): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): GroupNorm(1, 256, eps=1e-05, affine=False)
    (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=512, bias=False)
    )
    (3): ReLU(inplace=True)
    (4): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 256, eps=1e-05, affine=False)
    (6): Modulation2d(
      (layer): Linear(in_features=1024, out_features=512, bias=False)
    )
    (7): ReLU(inplace=True)
  )
  (skip): Identity()
)
(8): ResModConvBlock(
  (main): Sequential(
    (0): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): GroupNorm(1, 256, eps=1e-05, affine=False)
    (2): Modulation2d(
      (layer): Linear(in_features=1024, out_features=512, bias=False)
    )
    (3): ReLU(inplace=True)
    (4): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): GroupNorm(1, 256, eps=1e-05, affine=False)
```

```
      (6): Modulation2d(
        (layer): Linear(in_features=1024, out_features=512, bias=False)
      )
      (7): ReLU(inplace=True)
    )
    (skip): Identity()
  )
  (9): ResModConvBlock(
    (main): Sequential(
      (0): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
      (1): GroupNorm(1, 256, eps=1e-05, affine=False)
      (2): Modulation2d(
        (layer): Linear(in_features=1024, out_features=512, bias=False)
      )
      (3): ReLU(inplace=True)
      (4): Conv2d(256, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
      (5): GroupNorm(1, 128, eps=1e-05, affine=False)
      (6): Modulation2d(
        (layer): Linear(in_features=1024, out_features=256, bias=False)
      )
      (7): ReLU(inplace=True)
    )
    (skip): Conv2d(256, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
  )
  (10): Upsample(scale_factor=2.0, mode=bilinear)
 )
 (skip): Identity()
)
(5): ResModConvBlock(
 (main): Sequential(
  (0): Conv2d(256, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (1): GroupNorm(1, 128, eps=1e-05, affine=False)
  (2): Modulation2d(
    (layer): Linear(in_features=1024, out_features=256, bias=False)
  )
  (3): ReLU(inplace=True)
  (4): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (5): GroupNorm(1, 128, eps=1e-05, affine=False)
  (6): Modulation2d(
    (layer): Linear(in_features=1024, out_features=256, bias=False)
  )
  (7): ReLU(inplace=True)
 )
 (skip): Conv2d(256, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
)
(6): ResModConvBlock(
 (main): Sequential(
  (0): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (1): GroupNorm(1, 128, eps=1e-05, affine=False)
  (2): Modulation2d(
    (layer): Linear(in_features=1024, out_features=256, bias=False)
  )
  (3): ReLU(inplace=True)
  (4): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (5): GroupNorm(1, 128, eps=1e-05, affine=False)
  (6): Modulation2d(
    (layer): Linear(in_features=1024, out_features=256, bias=False)
  )
  (7): ReLU(inplace=True)
 )
 (skip): Identity()
)
(7): ResModConvBlock(
 (main): Sequential(
  (0): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (1): GroupNorm(1, 128, eps=1e-05, affine=False)
```

```
      (2): Modulation2d(
        (layer): Linear(in_features=1024, out_features=256, bias=False)
      )
      (3): ReLU(inplace=True)
      (4): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
      (5): GroupNorm(1, 128, eps=1e-05, affine=False)
      (6): Modulation2d(
        (layer): Linear(in_features=1024, out_features=256, bias=False)
      )
      (7): ReLU(inplace=True)
    )
    (skip): Identity()
  )
  (8): ResModConvBlock(
    (main): Sequential(
      (0): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
      (1): GroupNorm(1, 128, eps=1e-05, affine=False)
      (2): Modulation2d(
        (layer): Linear(in_features=1024, out_features=256, bias=False)
      )
      (3): ReLU(inplace=True)
      (4): Conv2d(128, 3, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
      (5): Identity()
      (6): Identity()
      (7): Identity()
    )
    (skip): Conv2d(128, 3, kernel_size=(1, 1), stride=(1, 1), bias=False)
  )
 )
)
```