💙

[5주차] Scalable Diffusion Models with Transformers

Abstract

DiTs(Diffusion Transformers): 이미지의 잠재적인 확산 모델을 훈련시키기 위해 U-Net 대신 **변환기 아키텍처**를 사용하는 새로운 클래스

- 확장성 분석: DiTs의 확장성은 전방 통과 복잡성(Gflops로 측정)을 통해 분석되었으며, 높은 Gflops를 가진 DiTs는 더 낮은 FID값을 가짐
  - DiTs에서는 변환기의 깊이나 너비를 증가시키거나 입력 토큰의 수를 늘림으로써 Gflops를 높일 수 있음
  - Gflops를 높인 DiTs는 더 낮은 FID 값을 가짐으로써, 이미지 생성의 품질이 향상됨
- 성능: DiT-XL/2 모델은 ImageNet 512×512 및 256x256 벤치마크에서 기존의 모든 확산 모델을 능가하는 최고의 성능을 달성했음

📷 **이미지의 잠재적 확산**

- **잠재적 확산 과정**: 주로 이미지를 생성하거나 수정할 때 사용되며, 이미지의 픽셀 공간이 아닌 **잠재 공간**(latent space)에서 작동함

-
잠재 공간: 이미지의 고차원 특성을 저차원으로 압축한 공간으로, 계산 효율성을 높이고 이미지 생성이나 수정을 더욱 정교하게 할 수 있도록 함

-
확산 모델(Diffusion Model): 이미지나 데이터를 점진적으로 변화시키는 과정을 모델링
1️⃣
순방향 확산: 데이터에 점차적으로 노이즈를 추가하는 과정
2️⃣
역방향 확산: 노이즈를 제거하며 원하는 이미지로 복원하는 과정

- 잠재적 확산 과정의 응용
1️⃣ 이미지 생성: 텍스트나 다른 이미지를 기반으로 새로운 이미지를 생성함
2️⃣ 이미지 변환: 기존 이미지를 수정하거나 스타일을 변환하는 데 사용됨

성능 지표

- **FID**(Fréchet Inception Distance): 생성된 이미지의 품질을 측정하는 지표로, **낮을수록 좋은 품질**을 의미함
- **Gflops**(초당 수행되는 부동 소수점 연산의 수): 모델의 전방 통과 복잡성을 측정하는 단위로, **높은 Gflops는 모델이 더 복잡하다**는 것을 의미함

Introduction

- U-Net의 도입
  - Ho 등의 연구에서 처음으로 확산 모델에 U-Net 아키텍처를 도입
  - U-Net 아키텍처는 초기에 픽셀 수준의 자동 회귀 모델과 조건부 GAN에서 성공을 거두었으며, 주로 ResNet 블록으로 구성됨
  - 트랜스포머에서 중요한 요소인 공간 자기 주의 블록을 추가하여 변형되었음

- Diffusion Transformers
  - 해당 논문에서는 트랜스포머 기반의 새로운 확산 모델 클래스인 DiTs에 대해 다룸
  - DiTs는 Vision Transformers(ViTs)을 따르며, 전통적인 CNN보다 시각적 인식에서 더 효과적으로 확장될 수 있음을 보여줌
    - DiTs가 이미지 토큰의 경로를 동적으로 조정하는 전략을 사용하여, 이미지 내의 중요한 정보를 더욱 효율적으로 처리할 수 있기 때문
    - ViTs의 기본 원리를 따르면서도 특정 작업에 더 적합한 방식으로 아키텍처를 확장함
  - 해당 연구는 네트워크의 복잡성 대비 샘플 품질의 상관관계를 보여주며, DiTs가 확산 모델에 대한 확장 가능한 아키텍처임을 입증함

Related Work

Transformers

트랜스포머의 다양한 적용 분야

- 언어 및 시각 분야
  - 트랜스포머는 자동 회귀 방식으로 픽셀을 예측하거나, 이산 코드북에 대해 훈련되어 자동 회귀 모델과 마스크 생성 모델로 사용됨
  - 특히 자동 회귀 모델은 최대 20B 파라미터까지 우수한 확장성을 보여줌
- DDPMs와 비공간 데이터 생성
  - 트랜스포머는 DDPMs(확산 확률 모델)에서 비공간 데이터, 예를 들어 DALL·E2에서 CLIP 이미지 임베딩을 생성하는 데 사용됨
  - DDPMs: 확산 확률 모델, 확산 과정과 역확산 과정으로 이루어짐
  - DALL·E2: 텍스트 설명을 바탕으로 이미지를 생성하는 OpenAI 모델로, 텍스트와 이미지 사이의 관계를 이해하고 텍스트 설명에 맞는 이미지를 생성할 수 있음
  - CLIP: 텍스트와 이미지를 연결하는 모델로, 이미지와 텍스트 사이의 의미적 관계를 학습함. DALL·E2에서는 CLIP을 통해 생성된 이미지 임베딩을 사용하여 텍스트 설명에 부합하는 이미지를 더 정확하게 생성할 수 있음

📂 **비공간 데이터 생성**

- 위치나 공간적 배치와 관련 없는 데이터를 생성하는 과정(↔**지리공간 데이터**: 지리적 위치 정보를 포함하는 데이터)

- 텍스트, 이미지, 소리 등 다양한 형태가 될 수 있음

- 비공간 데이터의 생성과 처리는 DDPMs과 같은 기술을 통해 이루어짐

- DALL·E 2와 같은 모델은 텍스트 설명을 바탕으로 이미지를 생성하는데, 이 과정에서 DDPMs가 활용되며 생성된 이미지는 위치 정보와 무관한 비공간 데이터의 예시라고 할 수 있음

Denoising diffusion probabilistic models (DDPMs)

확산 모델과 점수 기반 생성 모델의 성공

- 이미지 생성 분야의 성공
  - 확산 모델과 점수 기반 생성 모델은 이미지 생성 분야에서 GANs를 능가하는 성과를 보임
  - 이들 모델이 이미지 생성에 있어서 새로운 가능성을 열었음을 의미함
- DDPMs의 개선
  - **샘플링 기술의 개선**: DDPMs의 발전은 분류자 없는 가이드, 노이즈 예측 재구성, 캐스케이드 파이프라인 등 새로운 샘플링 기술의 도입으로 크게 발전했음
  - **U-Net 백본 아키텍처**: 모든 확산 모델에서 U-Net이 주요 백본 아키텍처로 사용되었으며, 이는 확산 모델의 표준 구조로 자리 잡았음
  - **트랜스포머 기반 아키텍처**: 최근 연구에서는 DDPMs를 위한 새로운 효율적인 아키텍처로 순수 트랜스포머를 소개하며, 이는 확산 모델의 발전 방향을 제시했음

Architecture complexity

- 아키텍처 복잡성 평가의 중요성
  - 파라미터 수의 한계
    - 파라미터 수는 이미지 모델의 복잡성을 평가하는 데 있어 한계가 있음
    - 특히 이미지 해상도와 같은 중요한 요소를 고려하지 않기 때문에, 성능에 대한 정확한 지표가 되지 못함

    - 계산량을 통한 분석
        - 본 논문에서는 아키텍처의 복잡성을 계산량을 통해 분석함
        - 아키텍처 설계 문헌에서 복잡성을 측정하는 표준 방법과 일치함
- 트랜스포머 클래스에 대한 초점
    - 트랜스포머의 적용: 해당 논문은 특히 **트랜스포머 아키텍처**를 기반으로 한 새로운 클래스의 확산 모델에 초점을 맞춤
    - 기존의 U-Net 아키텍처를 대체하는 새로운 접근 방식을 제시

3. Diffusion Transformers

3.1 Preliminaries

Diffusion formulation

- 전방 노이즈 과정
    - 실제 데이터 x0에 점진적으로 노이즈를 추가
    - **q(xt|x0) = N(xt; sqrt(α¯t)x0 + (1-α¯t)I)**
        - α¯t는 시간에 따라 변하는 하이퍼파라미터
        - sqrt(α¯t)x0는 원본 데이터 x0을 시간 t에 따라 조정하는 항
        - α¯t는 0과 1 사이의 값으로, 시간이 지남에 따라 점차 감소함 >> 원본 데이터의 정보를 점차 줄여가며 노이즈를 추가하는 역할
        - (1-α¯t)I는 추가되는 노이즈의 양을 조절, I는 단위 행렬이며, 1-α¯t는 시간에 따라 증가하는 값 >> 시간이 지날수록 더 많은 노이즈가 데이터에 추가됨을 의미함
        - N은 가우시안 분포를 의미하며, I는 단위 행렬
    - 재매개변화 기법: xt = sqrt(α¯t)x0 + sqrt(1-α¯t)εt
        - εt ~ N(0,I)는 표준 정규 분포에서 샘플링된 노이즈
        - 모델은 노이즈를 추가하는 과정을 학습할 수 있음

- 역방향 과정
    - 목표: 전방 과정에서 추가된 노이즈를 제거하여 원본 데이터를 복원함

    - **$p_\theta(x_{t-1}|x_t) = N(\mu_\theta(x_t), \Sigma_\theta(x_t))$**
        - 신경망은 μ_θ와 Σ_θ를 예측하는 데 사용됨
        - $\mu_\theta(x_t)$ 항은 모델이 예측하는 평균, θ는 모델의 파라미터를 나타내며 x_t는 현재 시점 t에서의 데이터를 의미 >> 결괏값인 평균은 다음 시점 t-1에서의 데이터 x_{t-1}의 위치를 예측하는 데 사용됨
        - $\Sigma_\theta(x_t)$ 항은 모델이 예측하는 공분산을 의미, 예측의 불확실성을 나타내며 데이터 x_{t-1}의 분포가 얼마나 넓게 퍼져있는지 나타냄
    - 훈련: 로그 가능도의 변분 하한을 최적화하여 모델을 훈련함
        - $L(\theta) = -p(x_0|x_1) + \Sigma_t D_{KL}(q^*(x_{t-1}|x_t, x_0)||p_\theta(x_{t-1}|x_t))$
        - $-p(x_0|x_1)$: 초기 데이터 x_0과 첫 번째 노이즈가 추가된 데이터 x_1 사이의 관계를 나타내는 로그 가능도, 모델이 얼마나 잘 초기 데이터를 복원할 수 있는지 평가함
        - $\Sigma_t D_{KL}$: 모든 시점 t에 대해 쿨백-라이블러 발산(KL divergence)의 합, 두 분포 사이(실제 데이터 분포, 예측 분포) 사이의 차이를 측정하며, 모델 학습의 목표는 이 값을 최소화하는 것
- 훈련 방법
    - 노이즈 예측 네트워크: $L_{simple}(\theta) = ||\varepsilon_\theta(x_t) - \varepsilon_t||^2$
        - 예측된 노이즈와 실제 노이즈 사이의 평균 제곱 오차를 최소화함
    - Nichol과 Dhariwal의 접근: ε_θ를 L_{simple}로 훈련시키고, Σ_θ를 전체 손실 L로 훈련시킴
- 이미지 샘플링
    - 새로운 이미지 생성: $x_{tmax} \sim N(0, I)$에서 시작하여 역방향 과정을 통해 x_{t-1}을 샘플링함

Classifier-free guidance

- Conditianal diffusion model: 클래스 레이블 c와 같은 추가 정보를 입력으로 사용
    - reverse process: $p_\theta(x_{t-1}|x_t, c)$
    - ϵ_θ와 Σ_θ는 c로 컨디셔닝됨
    - classifier-free guidance를 사용하여 샘플링 절차가 $\log p(c|x)$가 높은 x를 찾도록 장려할 수 있음 >> 주어진 조건 c에 대해 더 높은 확률을 가진 데이터 x를 생성

하려고 함

- 베이즈 정리
  - logp(c|x) ∝ logp(x|c) - logp(x): 조건부 확률과 베이즈 정리를 사용하여, 주어진 x에 대한 c의 확률을 계산함
  - ∇_x(logp(c|x)) ∝ ∇_x(logp(x|c)) - ∇_x(logp(x)): 조건부 확률의 기울기를 통해 어떻게 샘플링 방향을 조정할 수 있는지 보여줌
- DDPM 샘플링 절차
  - Diffusion model의 출력을 score function으로 해석하면 DDPM 샘플링 절차는 p(x|c)가 높은 샘플 x로 유도할 수 있음
  - $\epsilon_\theta(x_t,c)=\epsilon_\theta(x_t,\varnothing)+s\cdot\nabla_x\log p(x|c)\propto\epsilon_\theta(x_t,\varnothing)+s\cdot(\epsilon_\theta(x_t,c)-\epsilon\theta(x_t,\varnothing))$
    - $\epsilon_\theta(x_t,\varnothing)$에 $s\cdot\nabla_x\log p(x|c)$를 더함으로써, 조건 c에 따른 샘플링을 조정
  - s>1: guidance의 척도를 나타냄(모델이 얼마나 강하게 특정 조건을 따르도록 할 것인지를 결정)
  - c=ø으로 diffusion model을 평가하는 것은 학습 중에 c를 임의로 삭제하고 학습된 null 임베딩 ø으로 대체하여 수행됨

➡️ **Classifier-free guidance는 일반 샘플링 기술에 비해 상당히 개선된 샘플을 생성하는 것으로 널리 알려져 있으며, 이러한 추세는 DiT 모델에도 적용됨**

Latent diffusion models

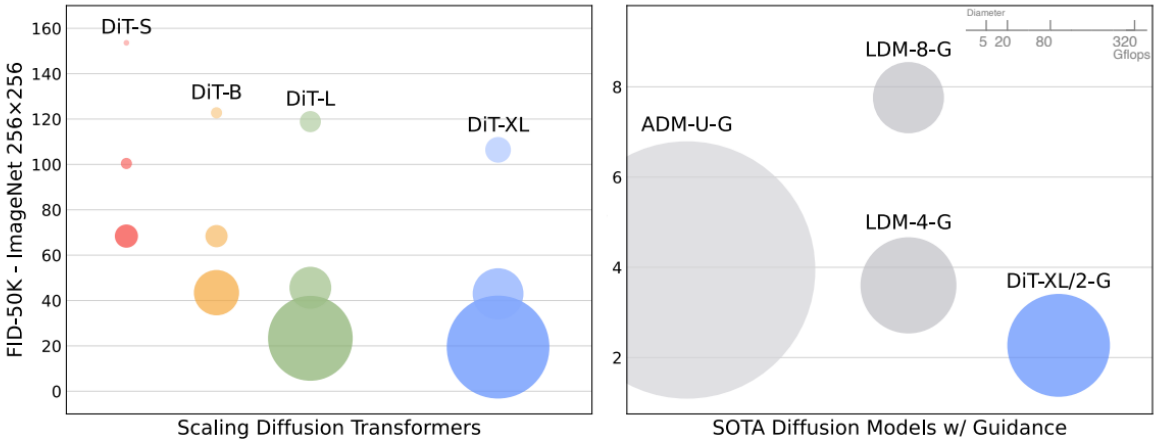

Figure 2: **ImageNet generation with Diffusion Transformers (DiTs).** Bubble area indicates the flops of the diffusion model. *Left:* FID-50K (lower is better) of our DiT models at 400K training iterations. Performance steadily improves in FID as model flops increase. *Right:* Our best model, DiT-XL/2, is compute-efficient and outperforms all prior U-Net-based diffusion models, like ADM and LDM.

- 고해상도 픽셀 space에서 직접 diffusion model을 학습하는 것은 불가능 할 수 있음
- Latent diffusion model은 2단계의 접근 방식으로 위 문제를 해결

1️⃣ 학습된 인코더 E를 사용하여, 이미지를 더 작은 space의 표현으로 압축하는 오토인코더를 학습함

2️⃣ 이미지 x의 diffusion model 대신, 표현 z = E(x)의 diffusion model을 학습함 (E는 고정됨)

- diffusion model에서 표현 z를 샘플링하고, 학습된 디코더 x=D(z)를 사용하여 이미지로 디코딩하여 새 이미지를 생성할 수 있음

🔎 참고: https://kimjy99.github.io/논문리뷰/dit/

3.2 Diffusion Transformer Design Space

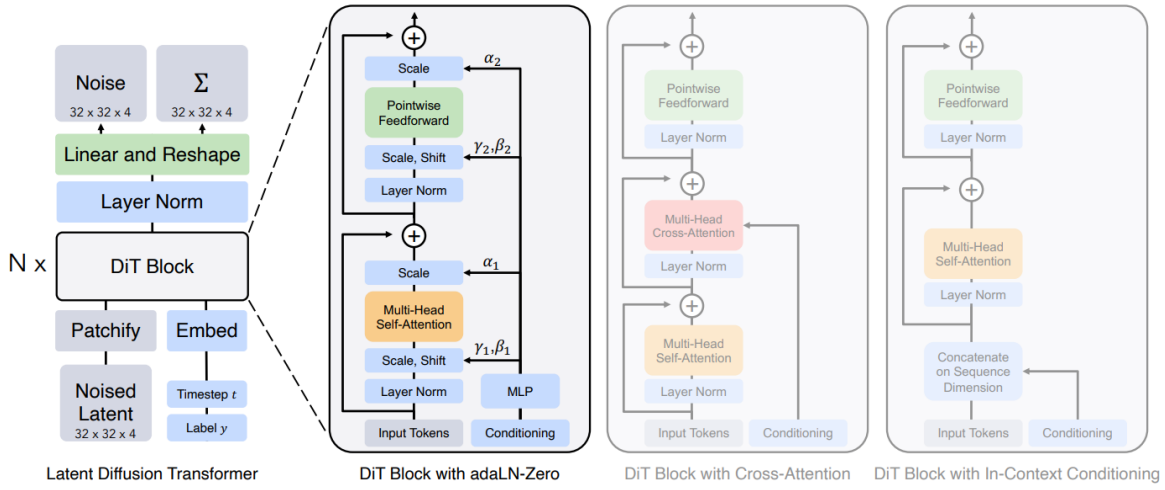

Figure 3: **The Diffusion Transformer (DiT) architecture.** *Left:* We train conditional latent DiT models. The input latent is decomposed into patches and processed by several DiT blocks. *Right:* Details of our DiT blocks. We experiment with variants of standard transformer blocks that incorporate conditioning via adaptive layer norm, cross-attention and extra input tokens. Adaptive layer norm works best.

- Diffusion Transformers (DiTs)의 핵심 요소
  - 트랜스포머 아키텍처: 표준 트랜스포머 아키텍처의 확장성 유지 + 이미지 생성에 특화된 새로운 구조를 제안
  - Vision Transformer (ViT) 기반: **이미지를 패치의 시퀀스로 처리**하는 ViT 아키텍처를 기반으로 함, 이미지의 공간적 표현을 효과적으로 학습
- Vision Transformer (ViT) 아키텍처
  - 이미지 처리 방식

    1️⃣ 이미지를 고정 크기의 패치로 분할

2️⃣ 각 패치를 선형적으로 임베딩

3️⃣ 위치 임베딩을 추가하여 표준 트랜스포머 인코더에 입력함

- 성능과 효율성
  - ViT는 컴퓨터 비전 분야에서 CNN을 대체할 수 있는 경쟁력으로 부상함
  - 계산 효율성과 정확도 면에서 현재의 최신 CNN 모델을 크게 앞서고 있음

Patchify: 이미지의 공간적 표현을 토큰 시퀀스로 변환

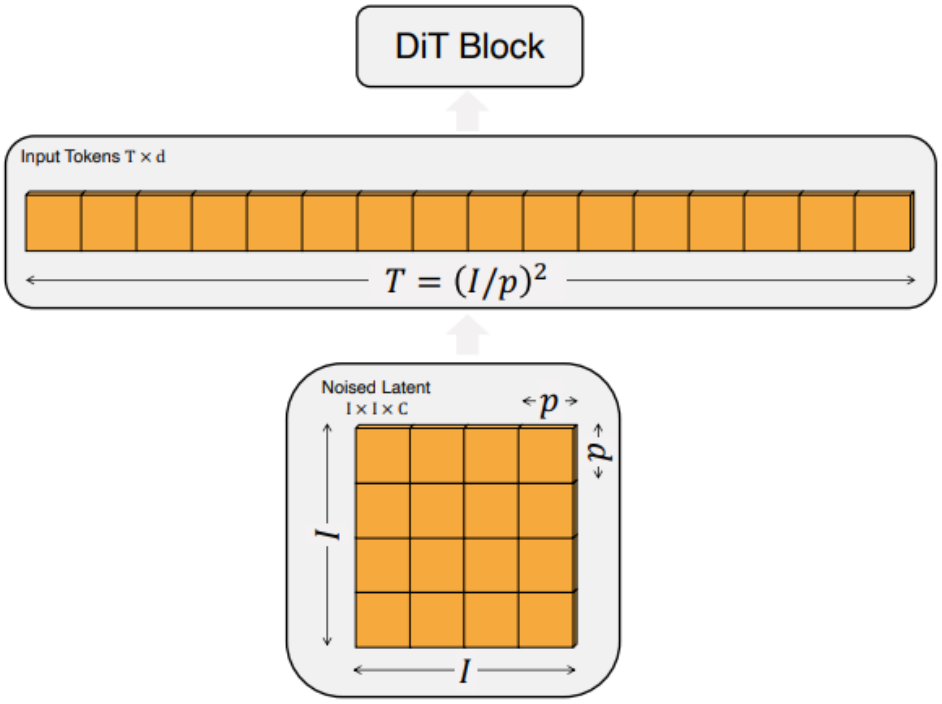

Figure 4: **Input specifications for DiT.** Given patch size $p \times p$, a spatial representation (the noised latent from the VAE) of shape $I \times I \times C$ is "patchified" into a sequence of length $T = (I/p)^2$ with hidden dimension d. A smaller patch size p results in a longer sequence length and thus more Gflops.

- Patchify 과정의 핵심 요소
  - 공간적 입력 변환: 입력 이미지의 공간적 표현을 T개의 토큰 시퀀스로 변환
  - 토큰 차원: 각 토큰은 d차원을 가짐

        - d가 클수록 각 토큰은 더 많은 정보를 담음 >> 세부 사항과 특성을 더 잘 포착
        - d가 클수록 모델의 계산 복잡성이 증가함
    - 패치 크기 하이퍼파라미터 (p): 패치의 크기를 결정하는 하이퍼파라미터로, (p)의 값에 따라 생성되는 토큰의 수 (T)가 결정됨
- Patchify의 영향
    - 패치 크기와 토큰 수: 패치 크기 (p)를 반으로 줄이면 토큰 수(T)는 4배 증가하며, 이는 트랜스포머의 총 Gflops를 최소 4배 증가시킴
    - Gflops에의 영향: 패치 크기를 변경하는 것은 Gflops에 큰 영향을 미치지만, 다운스트림 파라미터 수에는 의미 있는 영향을 주지 ❌
        - 다운스트림 파라미터: 모델이 학습된 후, 실제 세계의 다양한 작업(이미지 분류, 객체 감지 등)에 적용될 때 사용되는 파라미터
    - 디자인 공간에 추가된 p: (p=2, 4, 8)이 DiT 설계 공간(구성 요소와 하이퍼파라미터의 조합)에 추가됨 >> 각 패치 크기들이 모델의 성능과 효율성을 최적화하는 데 중요한 역할을 할 수 있음을 의미함

    ➡️ DiT는 표준 ViT의 주파수 기반 위치 임베딩(사인-코사인 버전)을 모든 입력 토큰에 적용하며, 이는 DiT가 이미지의 공간적 표현을 효과적으로 처리하고 이미지 생성 과정에서 높은 성능을 달성할 수 있도록 도움

DiT block design: 트랜스포머 아키텍처를 기반으로 한 새로운 클래스 모델, U-Net 백본을 트랜스포머로 대체하여 성능을 향상시키는 것이 목표.

입력 토큰은 일련의 Transformer block으로 처리되며 다음은 조건부 입력을 다르게 처리하는 Transformer block의 4가지 변형임

1️⃣ In-context conditioning

- 벡터 임베딩 추가: 노이즈 타임스탭 t와 클래스 라벨 c의 벡터 임베딩을 입력 시퀀스에 추가 >> ViTs의 cls 토큰과 유사하며, 표준 ViT 블록을 수정 없이 사용할 수 있도록 함
- 조절 토큰 제거: 최종 블록 후에는 조절 토큰을 시퀀스에서 제거 >> 모델에 미미한 새로운 Gflops를 도입

2️⃣ Cross-attention block

- 임베딩 연결: t와 c의 임베딩을 이미지 토큰 시퀀스와 별개의 길이가 두 개의 시퀀스로 연결함
- 크로스-어텐션 레이어 추가: 트랜스포머 블록은 멀티-헤드 자기주의 블록 다음에 추가 멀티-헤드 크로스-어텐션 레이어를 포함하도록 수정 >> 모델에 가장 많은 Gflops를 추

가, 대략 15%의 오버헤드를 추가

3️⃣ Adaptive layer norm, adaLN

- 레이어 노름 대체: 표준 레이어 노름 레이어를 적응형 레이어 노름으로 대체
- 임베딩 벡터 합에서 회귀: 차원별 스케일과 이동 매개변수 γ와 β를 직접 학습하는 대신, t와 c의 임베딩 벡터 합에서 이를 회귀함 >> 가장 적은 Gflops를 추가하며, 따라서 가장 계산 효율적임

4️⃣ adaLN-Zero 블록

- 항등 함수로 초기화: 각 레지듀얼 블록을 항등 함수로 초기화하는 것이 유익하다는 것을 발견
- Diffusion U-Net 모델은 유사한 초기화 전략을 사용하여 residual 연결 전에 각 블록의 최종 convolution layer를 0으로 초기화함
- 차원별 스케일링 매개 변수로 회귀: adaLN DiT 블록의 수정으로 γ와 β를 회귀할 뿐만 아니라, DiT 블록 내의 모든 레지듀얼 연결 바로 전에 적용되는 차원별 스케일링 매개변수 α로 회귀함 >> 전체 DiT 블록을 항등 함수로 초기화, 무시할 수 있는 Gflops를 모델에 추가

Model size

- 모델 구성
  - DiT-S, DiT-B, DiT-L, DiT-XL: 모델의 성능을 체계적으로 평가할 수 있는 다양한 크기를 제공, 각 구성은 N개의 DiT 블록 시퀀스를 적용하며 각 블록은 숨겨진 차원 크기 d에서 작동
  - 표준 트랜스포머 설정 사용: ViT(Vision Transformer)를 따라 N, d 및 주의 헤드를 함께 조정하는 표준 트랜스포머 구성을 사용
- 성능 평가
  - 다양한 flop 할당량: 0.3 Gflops에서 118.6 Gflops에 이르기까지 다양한 flop 할당량을 제공하여 모델의 크기에 따른 성능을 평가할 수 있도록 함
  - 확장성 평가: 다양한 모델 크기와 flop 할당량을 통해 DeT의 확장성과 성능을 체계적으로 평가할 수 있음

Transformer decoder

1️⃣ 출력 예측: 마지막 DiT 블록 후에는 이미지 토큰 시퀀스를 출력 노이즈 예측과 출력 대각 공분산 예측으로 디코딩함

2️⃣ 디코딩 과정: 표준 선형 디코더를 사용하여 최종 레이어 정규화 후, 각 토큰을 p×p×2C 텐서로 선형 디코딩함

3️⃣ 공간 레이아웃 재배열: 디코딩된 토큰을 원래의 공간 레이아웃으로 재배열하여 예측된 노이즈와 공분산을 얻음

4️⃣ 디자인 공간: DeT 디자인 공간은 패치 크기, 트랜스포머 블록 아키텍처, 모델 크기를 포함함

➡️ **DeT가 이미지 생성 과정에서 중요한 예측을 수행할 수 있도록 하며, 이미지 생성 분야에서 트랜스포머의 활용 가능성을 더욱 확장시키는 중요한 단계**

4. Experimental Setup

- DeT의 설계 공간을 탐색하고 모델 클래스의 스케일링 속성을 연구
- 모델은 구성(configs)과 잠재 패치 크기(p)에 따라 명명됨
  - DiT-XL/2는 XLarge 구성과 p=2를 의미

Training

- 모델 훈련
  - 클래스 조건부 잠재 DeT 모델을 ImageNet 데이터셋에서 256x256 및 512x512 이미지 해상도로 훈련
  - 해당 데이터셋은 경쟁이 치열한 생성 모델링 벤치마크임
- 초기화 및 가중치
  - 최종 선형 레이어를 0으로 초기화하고
  - ViT에서 사용되는 표준 가중치 초기화 기법을 사용
- 옵티마이저
  - 모든 모델은 AdamW를 사용하여 훈련됨
  - 학습률은 1x10^-4로 고정되며, 가중치 감소는 없고 배치 크기는 256으로 설정
- 데이터 증강
  - 유일하게 사용하는 데이터 증강은 수평 뒤집기
  - 학습률 웜업, 정규화 ❌
- 훈련 안정성
  - 훈련은 모든 모델 구성에서 매우 안정적이었으며, 트랜스포머 훈련 시 일반적으로 보이는 손실 급증을 관찰하지 못했음

- EMA
  - 훈련 중 DeT 가중치의 지수 이동 평균(EMA)을 0.9999의 감쇠로 유지
  - 논문에서 보고된 모든 결과는 EMA 모델을 사용
- 훈련 하이퍼파라미터
  - 모든 DeT 모델 크기와 패치 크기에 걸쳐 동일한 훈련 하이퍼파라미터를 사용
  - 훈련 하이퍼파라미터는 거의 ADM에서 그대로 유지됨
  - 학습률, 감쇠/웜업, Adam β1/β2, 가중치 감소 조정 ❌

Diffusion

- VAE 인코더
  - 주어진 RGB 이미지(x)가 256x256×3의 형태일 때, VAE 인코더는 다운샘플링 요소가 8인 z = E(x)로 변환하여 32x32×4의 형태로 만듦
- Z-공간에서의 작업
  - 모든 실험에서 디퓨전 모델은 이 Z-공간에서 작동함
  - 즉 이미지를 직접 다루는 대신, 이미지의 잠재 표현을 사용하여 작업함
- 새로운 잠재 변수 샘플링
  - 디퓨전 모델에서 새로운 잠재 변수를 샘플링한 후, VAE 디코더를 사용하여 이를 픽셀로 디코딩함(x = D(z))
- 디퓨전 하이퍼파라미터
  - ADM에서 가져온 디퓨전 하이퍼파라미터를 유지하며, 특히 tmax=1000의 선형 분산 일정을 사용함

Evaluation metrics

- 주요 평가 지표
  - Frechet Inception Distance (FID)
    - 이미지 생성 모델을 평가하는 표준 지표
    - 생성된 이미지와 실제 이미지 간의 거리를 측정하여, 모델이 얼마나 실제와 유사한 이미지를 생성하는지 평가
    - 해당 연구에서는 250 DDPM 샘플링 단계를 사용하여 FID-50K를 보고함
  - 구현 세부 사항의 민감성

        - FID는 작은 구현 세부 사항에 민감한 것으로 알려져 있음
        - 정확한 비교를 위해 이 논문에서 보고된 모든 값은 샘플을 내보내고 ADM의 TensorFlow 평가 스위트를 사용하여 얻은 것
    - 분류자 없는 가이드 사용
        - 해당 섹션에서 보고된 FID 수치는 특별히 명시되지 않는 한 분류자 없는 가이드를 사용하지 않음
- 보조 평가 지표
    - Inception Score (IS): 이미지가 얼마나 다양하고 명확한지를 평가
    - sFID: 특정 부분 집합에 대한 FID의 변형으로, 더 세분화된 평가를 제공
    - Precision/Recall: 생성된 이미지의 품질과 다양성을 평가하는 데 사용

Compute

- 모델 구현 및 훈련
    - JAX 사용
        - 모든 모델은 JAX라는 프로그래밍 언어로 구현되었음
        - AX는 고성능 수치 계산을 위해 설계된 언어로, 특히 머신러닝 연구와 개발에 널리 사용됨
    - TPU-v3 pods 사용
        - 모델 훈련에는 TPU-v3 pods가 사용되었음
        - TPU(Tensor Processing Unit)는 구글이 개발한 특수 하드웨어로, 머신러닝 작업을 위해 최적화되어있음
- DiT-XL/2 모델의 훈련 성능
    - TPU v3-256 pod에서의 훈련 속도
        - 가장 계산량이 많은 모델
        - TPU v3-256 pod를 사용하여 전역 배치 크기 256으로 대략 초당 5.7회의 반복(iterations)으로 훈련됨

5. Experiments

DiT block design

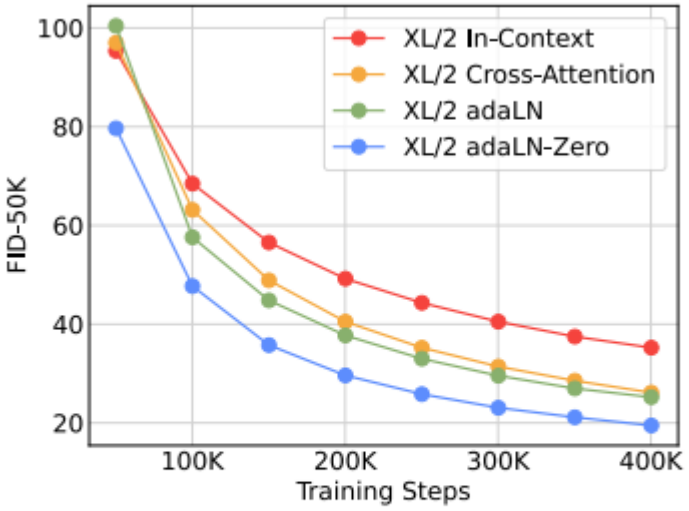

Figure 5: **Comparing different conditioning strategies.** adaLN-Zero outperforms cross-attention and in-context conditioning at all stages of training.

- DiT-XL/2 모델의 다양한 블록 디자인
  - 다양한 디자인의 모델
    - DiT-XL/2 모델의 네 가지 버전이 각각 다른 블록 디자인을 사용하여 훈련되었음
    - 인컨텍스트(in-context, 119.4 Gflops), 크로스-어텐션(cross-attention, 137.6 Gflops), 어댑티브 레이어 노말라이제이션(adaptive layer norm, adaLN, 118.6 Gflops), adaLN-zero(118.6 Gflops)
  - 성능 측정
    - 훈련 과정에서 FID(Fréchet Inception Distance)를 측정하여 모델의 성능을 평가
    - FID 점수는 생성된 이미지의 품질을 나타내는 지표로, 낮을수록 좋음
- 실험 결과
  - adaLN-Zero의 우수성
    - adaLN-Zero 블록을 사용한 모델이 크로스-어텐션 및 인컨텍스트 조건부 모델보다 낮은 FID를 달성
    - 가장 계산 효율적인 것으로 나타남
  - 400K 훈련 반복에서의 성과

        - adaLN-Zero 모델은 400K 훈련 반복 후, 인컨텍스트 모델의 FID의 거의 절반에 해당하는 성과를 보임
        - 조건부 메커니즘이 모델 품질에 중요한 영향을 미친다는 것을 보여줌
    - 초기화의 중요성
        - 모든 DiT 블록을 항등 함수로 초기화하는 adaLN-Zero가 일반 adaLN보다 훨씬 우수한 성능을 보임

Scaling model size and patch size

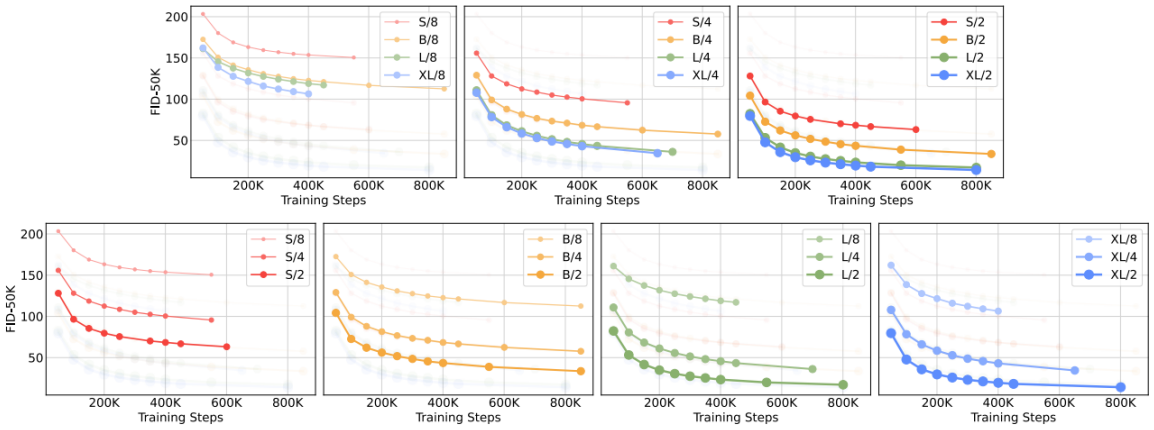

Figure 6: **Scaling the DiT model improves FID at all stages of training.** We show FID-50K over training iterations for 12 of our DiT models. *Top row:* We compare FID holding patch size constant. *Bottom row:* We compare FID holding model size constant. Scaling the transformer backbone yields better generative models across all model sizes and patch sizes.

- 모델 및 패치 크기의 다양성
    - 모델 구성: S, B, L, XL의 네 가지 모델 구성과 8, 4, 2의 세 가지 패치 크기를 가진 총 12개의 DiT 모델을 훈련시킴
    - Gflops와 FID: 각 모델의 Gflops(연산량)과 400K 훈련 반복 후의 FID(Fréchet Inception Distance)를 비교
        - FID는 낮을수록 좋으며, 이미지의 품질을 나타냄
- 실험 결과
    - 모델 크기 증가의 효과
        - 모델 크기를 증가시키고 패치 크기를 감소시키면 확산 모델의 성능이 크게 향상됨
    - 모델 크기에 따른 FID 변화

        - 모델 크기를 증가시키면서 패치 크기를 일정하게 유지했을 때, 모든 구성에서 훈련의 모든 단계에서 FID가 크게 개선됨
        - 트랜스포머를 더 깊고 넓게 만들어 성능을 향상시킬 수 있음을 보여줌
    - 패치 크기 감소의 효과
        - 패치 크기를 감소시키면서 모델 크기를 일정하게 유지했을 때, 훈련 동안 FID가 크게 개선되었음
        - 이는 DiT가 처리하는 토큰의 수를 단순히 늘림으로써 성능을 향상시킬 수 있음을 의미함

DiT Gflops are critical to improving performance

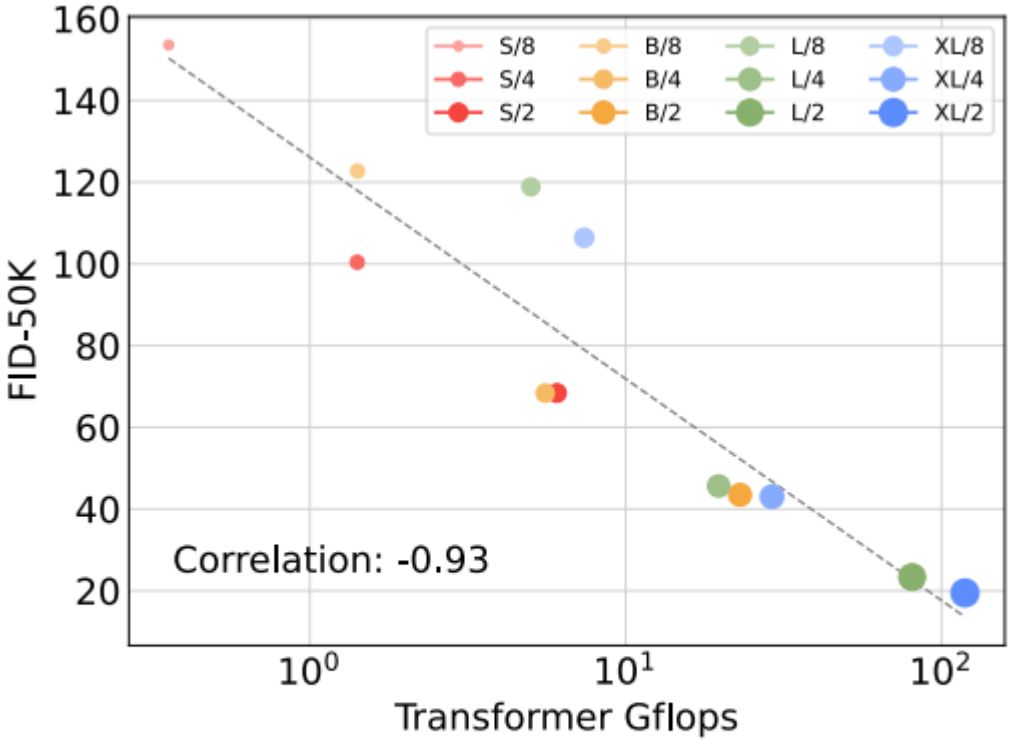

Figure 8: **Transformer Gflops are strongly correlated with FID.** We plot the Gflops of each of our DiT models and each model's FID-50K after 400K training steps.

- Gflops와 모델 품질의 관계
    - 파라미터 수와 품질

        - 모델의 크기를 일정하게 유지하고 패치 크기를 줄이면 트랜스포머의 총 파라미터 수는 크게 변하지 않음
        - 파라미터 수가 DiT 모델의 품질을 결정하는 유일한 요소가 아님을 시사함
    - Gflops의 증가
        - 패치 크기를 줄이면 Gflops만 증가
        - 모델의 Gflops를 증가시키는 것이 실제로 성능 향상의 핵심임을 나타냄
- Gflops와 FID의 관계
    - FID-50K와 Gflops의 비교
        - 400K 훈련 단계에서 FID-50K와 모델 Gflops를 비교한 결과, 총 Gflops가 비슷한 다양한 DiT 구성이 비슷한 FID 값을 얻는 것으로 나타남
    - Gflops와 FID-50K의 상관관계
        - 모델 Gflops와 FID-50K 사이에는 강한 음의 상관관계가 있으며, 이는 추가적인 모델 연산이 DiT 모델의 성능 향상에 있어 중요한 요소임을 시사함

Larger DiT models are more compute-efficient

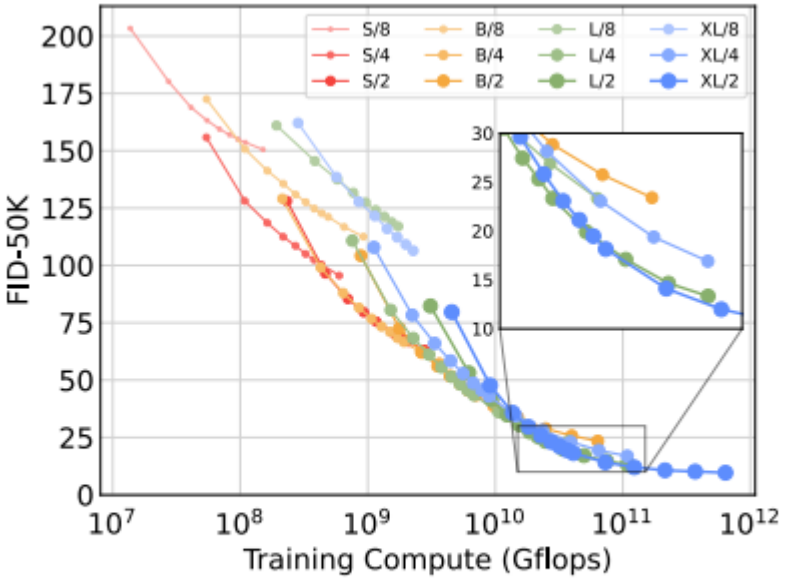

Figure 9: **Larger DiT models use large compute more efficiently.** We plot FID as a function of training compute.

- 훈련 계산량의 추정

    - 계산량 추정 방법: 모델 Gflops, 배치 크기, 훈련 단계, 그리고 역전파가 순전파보다 약 2배 더 많은 계산을 필요로 한다는 가정(계수 3)을 사용하여 훈련 계산량을 추정함
- 크고 작은 모델의 비교
    - 작은 DiT 모델: 작은 모델을 더 오래 훈련시키더라도, 결국에는 적은 훈련 단계로 훈련된 큰 모델에 비해 계산 비효율적이 됨
    - 패치 크기의 영향: 패치 크기만 다른 모델들도 훈련 Gflops를 통제했을 때 서로 다른 성능 프로필을 보임
        - XL/4는 약 10^10 Gflops 후에 XL/2에 의해 성능이 능가됨

Visualizing scaling

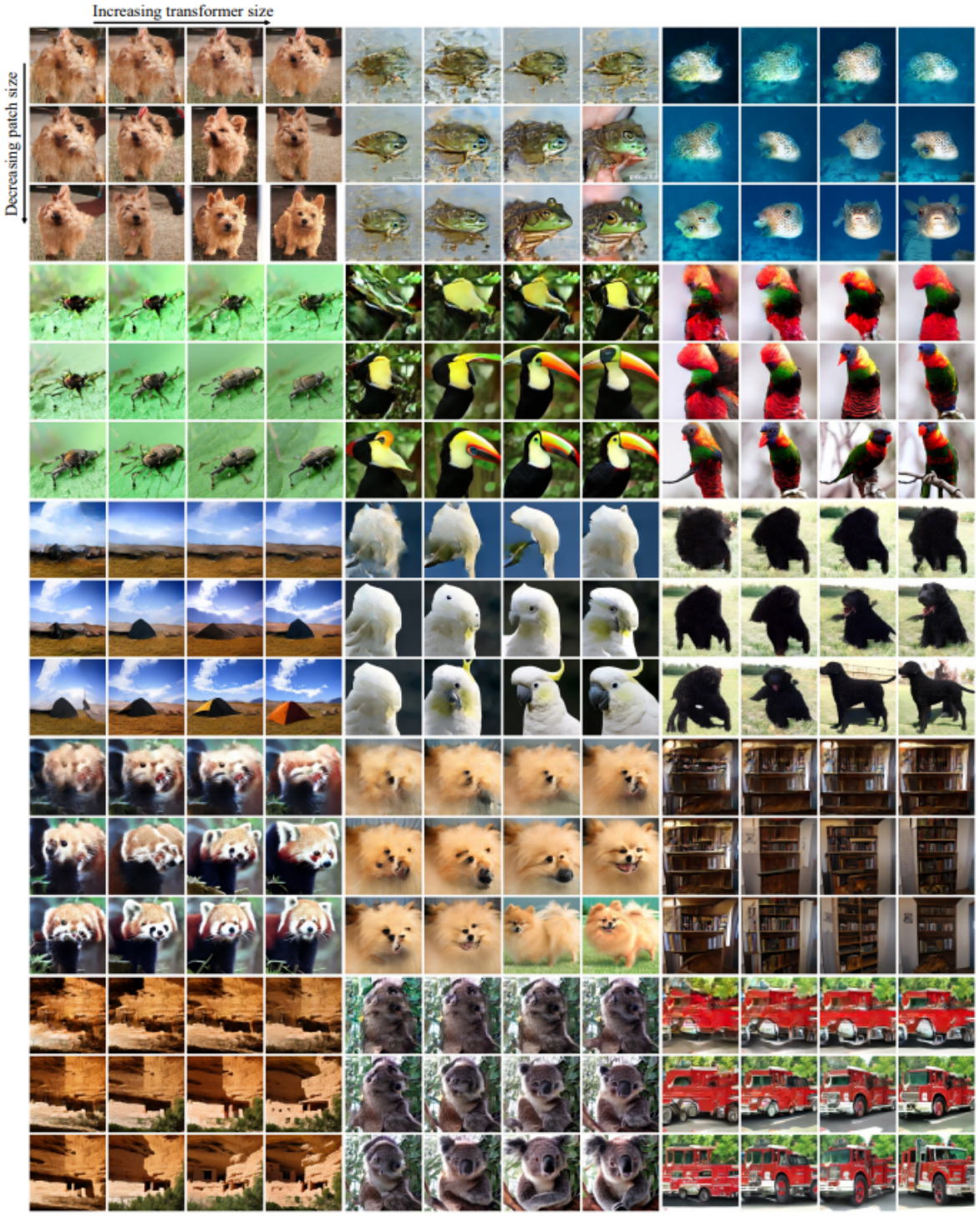

Figure 7: **Increasing transformer forward pass Gflops increases sample quality.** *Best viewed zoomed-in.* We sample from all 12 of our DiT models after 400K training steps using the same input latent noise and class label. Increasing the Gflops in the model—either by increasing transformer depth/width or increasing the number of input tokens—yields significant improvements in visual fidelity.

- 스케일링의 효과
  - 모델 크기와 토큰 수 증가: 모델의 크기와 토큰 수를 동시에 증가시키는 스케일링은 시각적 품질에서 눈에 띄는 개선을 가져옴

State-of-the-Art Diffusion Models - 256*256 ImageNet

Class-Conditional ImageNet 256×256					
Model	FID↓	sFID↓	IS↑	Precision↑	Recall↑
BigGAN-deep [2]	6.95	7.36	171.4	0.87	0.28
StyleGAN-XL [53]	2.30	4.02	265.12	0.78	0.53
ADM [9]	10.94	6.02	100.98	0.69	0.63
ADM-U	7.49	5.13	127.49	0.72	0.63
ADM-G	4.59	5.25	186.70	0.82	0.52
ADM-G, ADM-U	3.94	6.14	215.84	0.83	0.53
CDM [20]	4.88	-	158.71	-	-
LDM-8 [48]	15.51	-	79.03	0.65	0.63
LDM-8-G	7.76	-	209.52	0.84	0.35
LDM-4	10.56	-	103.49	0.71	0.62
LDM-4-G (cfg=1.25)	3.95	-	178.22	0.81	0.55
LDM-4-G (cfg=1.50)	3.60	-	247.67	**0.87**	0.48
DiT-XL/2	9.62	6.85	121.50	0.67	**0.67**
DiT-XL/2-G (cfg=1.25)	3.22	5.28	201.77	0.76	0.62
DiT-XL/2-G (cfg=1.50)	**2.27**	**4.60**	**278.24**	0.83	0.57

Table 2: **Benchmarking class-conditional generation on ImageNet 256×256.** DiT achieves state-of-the-art FID.

- 주요 성능 지표
  - FID 점수
    - DiT-XL/2는 분류기 없는 가이드를 사용하여 모든 이전 생성 모델들 중 가장 낮은 FID 점수를 달성함
    - 이전의 최고 기록인 StyleGAN-XL을 포함한 모든 이전 모델들을 능가하는 것
  - 계산 효율성
    - DiT-XL/2는 계산 효율성 측면에서도 우수함을 보여주며, 특히 잠재 공간 및 픽셀 공간 U-Net 모델들과 비교했을 때 눈에 띄는 차이를 보임
  - 리콜 값
    - DiT-XL/2는 테스트된 모든 분류기 없는 가이드 스케일에서 LDM-4 및 LDM-8보다 높은 리콜 값을 달성함

State-of-the-Art Diffusion Models - 512*512 ImageNet

Class-Conditional ImageNet 512×512					
Model	FID↓	sFID↓	IS↑	Precision↑	Recall↑
BigGAN-deep [2]	8.43	8.13	177.90	0.88	0.29
StyleGAN-XL [53]	2.41	4.06	267.75	0.77	0.52
ADM [9]	23.24	10.19	58.06	0.73	0.60
ADM-U	9.96	5.62	121.78	0.75	**0.64**
ADM-G	7.72	6.57	172.71	**0.87**	0.42
ADM-G, ADM-U	3.85	5.86	221.72	0.84	0.53
DiT-XL/2	12.03	7.12	105.25	0.75	**0.64**
DiT-XL/2-G (cfg=1.25)	4.64	5.77	174.77	0.81	0.57
DiT-XL/2-G (cfg=1.50)	**3.04**	**5.02**	**240.82**	0.84	0.54

Table 3: **Benchmarking class-conditional image generation on ImageNet 512×512.** Note that prior work [9] measures Precision and Recall using 1000 real samples for 512 × 512 resolution; for consistency, we do the same.

- 성능 개선: DiT-XL/2는 512x512 해상도에서 ADM이 달성한 이전 최고 FID를 3.04로 개선하며 모든 이전 확산 모델을 능가함
- 계산 효율성: 증가된 토큰 수에도 불구하고, XL/2는 524.6 Gflops만을 사용하여 계산 효율성을 유지함

Scaling Model vs. Sampling Compute

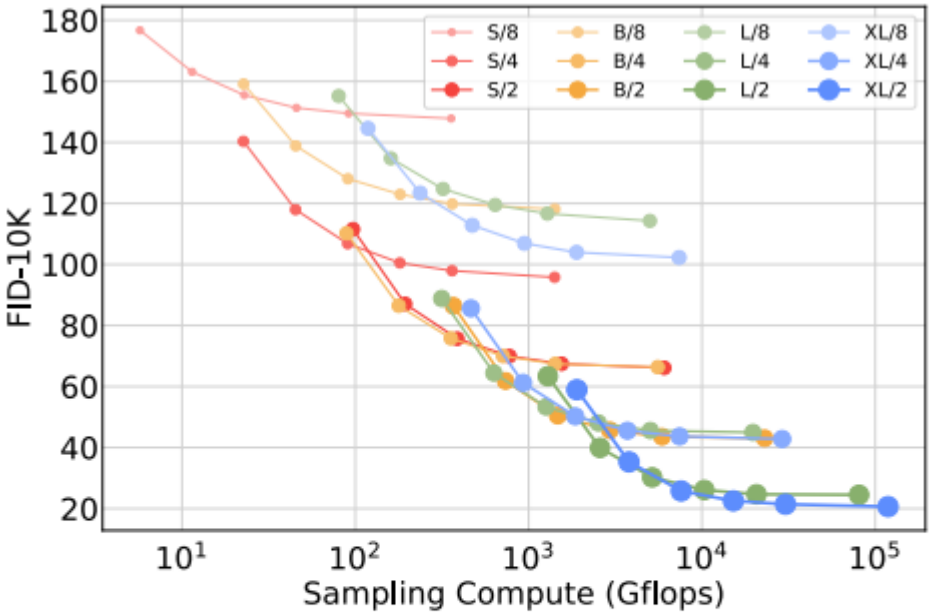

Figure 10: **Scaled-up *sampling* compute does not compensate for a lack of *model* compute.** We plot FID-10K of DiT models after 400K training iterations using [16, 32, 64, 128, 256, 1000] sampling steps against the total compute used to sample each image. Small models cannot close the performance gap with our large models, even if they sample with more test-time Gflops than the large models.

- 샘플링 단계와 FID: 더 많은 샘플링 단계를 사용하는 것이 반드시 더 나은 이미지 품질로 이어지지는 않음
- 계산 효율성: 더 큰 모델(DiT-XL/2)이 적은 계산량으로도 더 작은 모델(DiT-L/2)보다 더 나은 FID 점수를 달성할 수 있음
- 모델 크기의 중요성: 샘플링 계산을 늘리는 것은 모델의 계산 능력 부족을 보완할 수 없음을 시사함

6. Conclusion

- DiTs의 주요 장점
  - 성능 향상: DiTs는 U-Net 기반 모델에 비해 더 나은 성능, 유연성, 그리고 효율성을 제공함
  - 확장성: DiTs는 트랜스포머 아키텍처를 기반으로 하여 뛰어난 확장성을 가지며, 이는 더 큰 모델과 더 많은 토큰 수로의 확장 가능성을 의미함

    - 적용 가능성: DiT는 텍스트-이미지 생성 모델에 대한 백본으로 사용될 수 있는 잠재력을 가지고 있으며, 이는 DALL·E 2와 Stable Diffusion과 같은 모델에 적용될 수 있음
- 결론 및 향후 방향
    - DiTs는 U-Net 모델을 능가하는 성능과 함께 트랜스포머 모델의 확장성을 제공함

      ➡️ **DiTs는 다양한 확산 모델링 작업에 있어서 매우 유망한 선택**
    - DiTs의 확장성은 더 큰 모델과 토큰 수로의 확장을 가능하게 하며, 이는 향후 연구의 중요한 방향이 될 것

논문에 대한 의견 및 의문점(꼭지)

➡️ 확장 가능한 확산 모델을 개발할 때, AI가 생성한 이미지가 사회적, 윤리적 문제를 야기할 수 있는 경우, 이를 감지하고 관리할 수 있는 대책이 필요할 것으로 생각됨. 또한 대규모 데이터셋에서 학습될 때 개인 정보 보호 및 데이터 프라이버시에 대한 우려도 필요하다고 봄