

Difüzyon Modelleri ve Stable Diffusion

Gürültüden görüntüye: teoriden uygulamaya

İleri Yapay Zekâ — Lisansüstü Ders Notu

Şırnak Üniversitesi

Bu derste nereye gidiyoruz?

Üç soruyu cevaplayacağız:

1. Bir görüntüyü **kontrollü biçimde** nasıl bozabiliriz?
2. Bu bozulmayı **geri almayı** bir sinir ağına nasıl öğretiriz?
3. Aynı fikir, 512×512 görüntülerde nasıl **Stable Diffusion**'a dönüşür?

Her bölümün sonunda uygulama notebook'undaki ilgili TODO numarası verilir.
Teoriyi burada görüp kodu orada yazacaksınız.

Bölüm	Konu
1	İleri süreç (bozma)
2	Gürültü takvimleri
3	Gürültü sökücü: U-Net
4	Eğitim
5	Örnekleme: DDPM / DDIM
6	Koşullu üretim ve CFG
7	Latent difüzyon, Stable Diffusion

Motivasyon

Üretken modeller manzarası

	GAN	VAE	Normalizing Flow	Difüzyon
Eğitim	min-max, kırılğan	kararlı	kararlı	çok kararlı
Örnek kalitesi	yüksek	bulanık	orta	çok yüksek
Mod kapsamı	zayıf (mod çökmesi)	iyi	iyi	iyi
Üretim hızı	1 geçiş	1 geçiş	1 geçiş	10-1000 geçiş
Olabilirlik	yok	alt sınır	tam	alt sınır

GAN = Generative Adversarial Network (çekişmeli üretici ağ) · **VAE** = Variational Autoencoder (değişimsel otokodlayıcı) · *mod çökmesi* (mode collapse) = modelin verinin yalnızca birkaç tipini üretmesi · *olabilirlik* (likelihood) = modelin veriye atadığı olasılık

Difüzyonun takası

Zor olan *tek bir* problemi (gürültüden görüntüye tek adımda atlamak), kolay olan *yüzlerce* probleme böler. Bedeli: üretim yavaştır.

Masanın üstünde kumla yapılmış bir resim düşünün. Masayı hafifçe sarsarsanız resim biraz bozulur. Sarsmaya devam ederseniz elinizde **rastgele kum** kalır.

Difüzyonun tek cümlelik özeti

Bozma işleminin matematiği tam olarak biliniyorsa, bir ağa bu sürecin **tersini** öğretebiliriz. Sonra rastgele kumdan başlayıp geri sararak **yeni** bir resim üretebiliriz.

Neden işe yarıyor?

- Bozma süreci *sabit* ve *parametresiz*: öğrenilecek bir şey yok.
- Her adım *küçük*: ağın işi kolay bir regresyon.
- Hedef hep aynı dağılımdan: $\varepsilon \sim \mathcal{N}(0, \mathbf{I})$.

Bu ders notundaki tüm sonuçlar 28×28 gri tonlu, dört sınıflı küçük bir kıyafet silueti kümesi üzerinde üretildi. Küçük olması kasıtlı: eğitim birkaç dakikada bitiyor, dolayısıyla derste canlı denenebiliyor.



Görüntüler $[-1, 1]$ aralığına ölçeklenir. Sebebi birazdan netleşecek: gürültümüz $\mathcal{N}(0, 1)$ 'den geliyor, verinin de benzer ölçekte olması işi kolaylaştırıyor.

Bölüm 1 — İleri Süreç

Bozma süreci: tek adım

Her adımda görüntüyü birazcık küçültüp üzerine birazcık **Gauss gürültüsü** (normal dağılımdan çekilen rastgele değerler) ekliyoruz:

$$q(x_t \mid x_{t-1}) = \mathcal{N}(x_t; \sqrt{1 - \beta_t} x_{t-1}, \beta_t \mathbf{I})$$

- $\beta_t \in (0, 1)$: **gürültü takvimi**. t büyüdükçe artar — sona doğru daha sert sarsarız.
- $\sqrt{1 - \beta_t}$ çarpanı varyansın patlamasını engeller: x_{t-1} birim varyanslıysa x_t de öyle kalır.
- Bu bir **Markov zinciri**: her adım yalnızca bir öncekine bağlıdır. $x_0 \rightarrow x_1 \rightarrow \dots \rightarrow x_T$.

Sorun

$t = 200$ 'deki görüntüyü istiyorsak 200 adım tek tek mi uygulayacağız? Eğitimde bu kabul edilemez.

Kapalı form: tek hamlede herhangi bir t 'ye

İki bağımsız Gauss'un toplamı yine Gauss'tur ve varyansları toplanır. Bunu zincir boyunca uygularsak, $\alpha_t = 1 - \beta_t$ ve $\bar{\alpha}_t = \prod_{s=1}^t \alpha_s$ tanımıyla:

İleri sürecin kapalı formu

$$q(x_t | x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t} x_0, (1 - \bar{\alpha}_t)\mathbf{I}) \quad \Longleftrightarrow \quad x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} \varepsilon$$

$$\varepsilon \sim \mathcal{N}(0, \mathbf{I}) \quad (\text{yeniden parametreleme hilesi, reparameterization trick})$$

- $\bar{\alpha}_t$, 1'den 0'a düşer: resmin katsayısı söner, gürültünün katsayısı büyür.
- $\bar{\alpha}_t$ aslında **sinyal/gürültü oranının** (SNR, signal-to-noise ratio) kendisidir: $\text{SNR}(t) = \bar{\alpha}_t / (1 - \bar{\alpha}_t)$.
- Artık eğitimde rastgele bir t seçip *tek satırda* x_t üretebiliriz.

Neden iki Gauss birleşince tek Gauss?

Ardışık iki adımı açalım ($\varepsilon_1, \varepsilon_2$ bağımsız):

$$\begin{aligned}x_t &= \sqrt{\alpha_t} x_{t-1} + \sqrt{1 - \alpha_t} \varepsilon_1 \\&= \sqrt{\alpha_t} \left(\sqrt{\alpha_{t-1}} x_{t-2} + \sqrt{1 - \alpha_{t-1}} \varepsilon_2 \right) + \sqrt{1 - \alpha_t} \varepsilon_1 \\&= \sqrt{\alpha_t \alpha_{t-1}} x_{t-2} + \underbrace{\sqrt{\alpha_t (1 - \alpha_{t-1})} \varepsilon_2 + \sqrt{1 - \alpha_t} \varepsilon_1}_{\text{varyansı } \alpha_t - \alpha_t \alpha_{t-1} + 1 - \alpha_t = 1 - \alpha_t \alpha_{t-1}}\end{aligned}$$

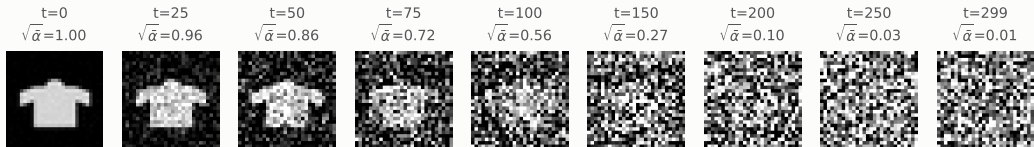
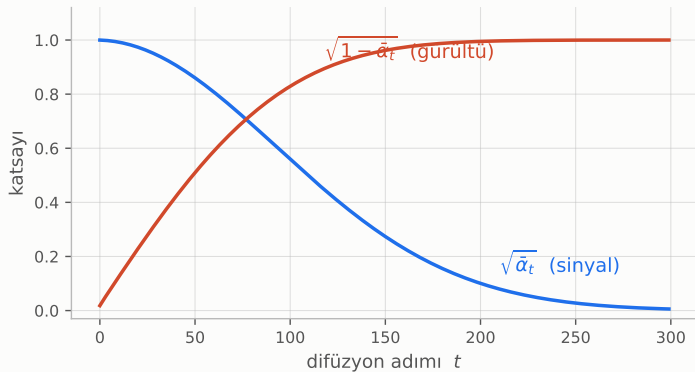
Yani iki adımın bileşkesi, α 'ların *çarpımıyla* tanımlı tek bir adıma eşdeğer.

Tümevarımla t adım için $\bar{\alpha}_t = \prod_s \alpha_s$ çıkar.

Ders

Difüzyonun pratik olmasının sebebi tam olarak budur. Bu özellik olmasaydı her eğitim adımı $O(T)$ maliyetli olurdu.

Gözle görelim



Hangi t zor?

- t **küçük**: görüntü neredeyse temiz. Ama model, resmin *altına* saklanmış ince gürültüyü bulmak zorunda — aslında zor.
- t **büyük**: elde zaten gürültü var. “Girdinin kendisi” bile iyi bir tahmin — kolay.
- **Ortada**: hem yapı hem gürültü var. Öğrenmenin asıl gerçekleştiği bölge.

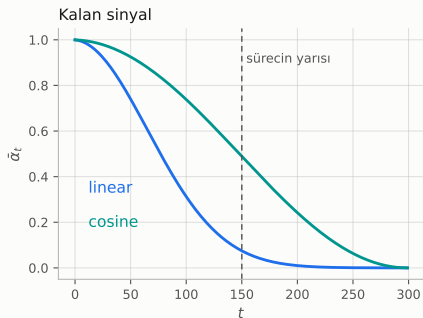
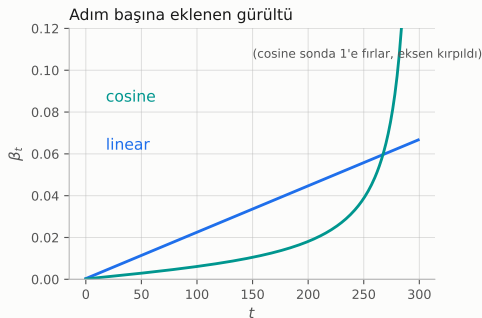
Pratik sonuç

Eğitim bütçesini t 'ler arasında nasıl dağıttığınız önemlidir. Düz tekdüze (uniform) örnekleme en basit seçenektir; literatürde önem örnekleme (importance sampling) ve $\min$ -SNR ağırlıklandırma gibi iyileştirmeler vardır.

Bölüm 2 — Gürültü Takvimleri

Lineer takvimin sorunu

DDPM (Denoising Diffusion Probabilistic Models — gürültü gidermeli olasılıksal difüzyon modeli) makalesinin orijinal seçimi: β_t , 10^{-4} 'ten 0,02'ye lineer ($T = 1000$).



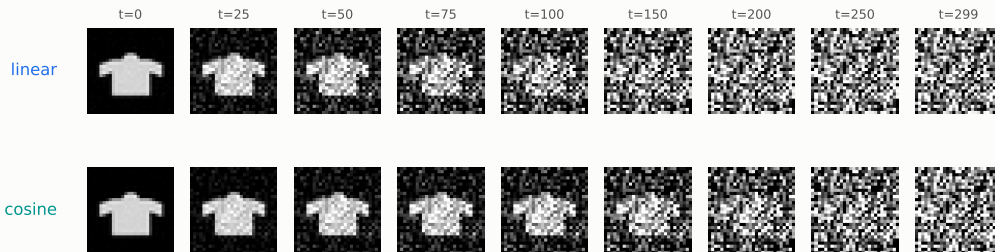
Sürecin yarısında ($t = T/2$) lineer takvimde $\bar{\alpha} \approx 0,08$ kalmış — görüntü çoktan ölmüş. Eğitim bütçesinin yarısı **gürültüden gürültü tahmin etmeye** harcanıyor.

Kosinüs takvimi (Nichol & Dhariwal, 2021)

β_t 'yi doğrudan tanımlamak yerine $\bar{\alpha}_t$ 'yi tanımlayıp β_t 'yi geri çıkarıyorlar:

$$\bar{\alpha}_t = \frac{f(t)}{f(0)}, \quad f(t) = \cos^2\left(\frac{t/T + s}{1 + s} \cdot \frac{\pi}{2}\right), \quad \beta_t = 1 - \frac{\bar{\alpha}_t}{\bar{\alpha}_{t-1}}$$

$s = 0,008$ küçük bir kaydırmadır: $t \approx 0$ civarında β 'nin sıfıra çok yaklaşıp sayısal sorun çıkarmasını engeller. β_t pratikte 0,999 ile kırpılır.



Takvim seçimi neden önemli?

Kosinüsün kazancı

Aynı eğitim bütçesiyle daha çok *bilgilendirici* adım; dolayısıyla daha iyi örnek kalitesi. Özellikle düşük çözünürlükte fark belirgindir.

Tuzak

Farklı takvimlerin kayıp değerleri **doğrudan karşılaştırılamaz**: farklı takvim, farklı zorlukta bir problem demektir. Düşük kayıp her zaman daha iyi örnek anlamına gelmez.

Diğer seçenekler

- **sigmoid / karesel** takvimler
- **Zero terminal SNR**: $\bar{\alpha}_T$ 'yi tam olarak 0 yapma düzeltmesi (sonda sinyalin tamamen silinmesi) — aksi halde model asla tam siyah/tam beyaz görüntü üretemez.
- **v-parametrelemesi**: ε yerine $v = \sqrt{\bar{\alpha}} \varepsilon - \sqrt{1 - \bar{\alpha}} x_0$ tahmini; az adımlı üretimde daha kararlı.

Notebook: TODO-3 (kosinüs takvimi) ve Bölüm 8 deney laboratuvarı.

Bölüm 3 — Gürültü Sökücü

Modelden tam olarak ne isteyeceğiz?

Üç eşdeğer seçenek var. Model şunlardan birini tahmin edebilir:

Parametreleme	Hedef	Yorum
x_0 - tahmini	temiz görüntü	büyük t 'de hedef neredeyse öngörülemez
ε - tahmini	eklenen gürültü	hedefin istatistiği her t 'de sabit
v - tahmini	$\sqrt{\bar{\alpha}} \varepsilon - \sqrt{1 - \bar{\alpha}} x_0$	ikisinin melezi; az adımda kararlı

Biz ε -tahmini kullanacağız. Birini bilerseniz diğerini formülle çıkarabilirsiniz:

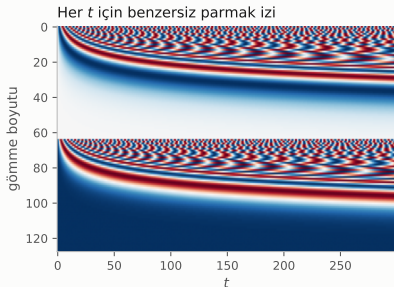
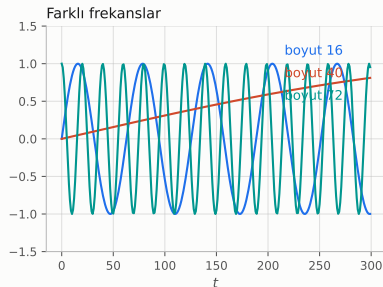
$$\hat{x}_0 = \frac{x_t - \sqrt{1 - \bar{\alpha}_t} \varepsilon_\theta(x_t, t)}{\sqrt{\bar{\alpha}_t}}$$

Neden ε ? Hedef her zaman $\mathcal{N}(0, \mathbf{I})$ 'dan geldiği için kayıp ölçüğü t 'den bağımsızdır — bu, eğitimi kayda değer biçimde stabilize eder.

Zaman bilgisini ağa nasıl veririz?

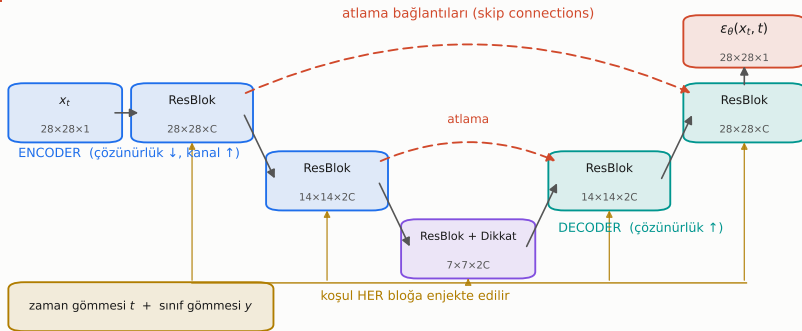
Bir tamsayıyı ağa olduğu gibi vermek kötü bir fikirdir: ağ ölçeği öğrenmekte zorlanır. Transformer'lardaki sinüzoidal konum kodlamasını ödünç alıyoruz:

$$\text{emb}(t)_i = \sin\left(\frac{t}{10000^{2i/d}}\right), \quad \text{emb}(t)_{i+d/2} = \cos\left(\frac{t}{10000^{2i/d}}\right), \quad i = 0, \dots, \frac{d}{2} - 1$$



Yakın t 'ler benzer vektör alır, uzak t 'ler farklı. Notebook: TODO-4.

Mimari: neden U-Net?



U-Net adını yukarıdaki şemanın U harfine benzeyen biçiminden alır. Girdi ve çıktı aynı boyutta: bu bir *görüntüden görüntüye* problemi.

- Encoder: çözünürlük ↓, kanal ↑ — küresel yapı.
- Decoder: çözünürlük ↑ — piksel düzeyinde çıktı.

- **Atlama bağlantıları**: ince detay doğrudan taşınır — çıktının girdiyle piksel piksel hizalı olması gerekir.
- **Dikkat katmanı**: en düşük çözünürlükte uzak pikseller konuşur.
- **Koşul** (t , sınıf) her bloğa kanal başına kaydırma olarak girer.

Bölüm 4 — Eğitim

Kayıp fonksiyonu nereden geliyor?

Model, veri olabilirliğinin **değişimsel alt sınırını** (ELBO, *evidence lower bound*) maksimize eder. Zincir boyunca açıldığında ELBO üç parçaya ayrılır:

$$\underbrace{D_{\text{KL}}(q(x_T|x_0) \parallel p(x_T))}_{\text{sabit, parametresiz}} + \sum_{t>1} \underbrace{D_{\text{KL}}(q(x_{t-1}|x_t, x_0) \parallel p_{\theta}(x_{t-1}|x_t))}_{\text{asıl terim}} - \underbrace{\log p_{\theta}(x_0|x_1)}_{\text{yeniden kurulum}}$$

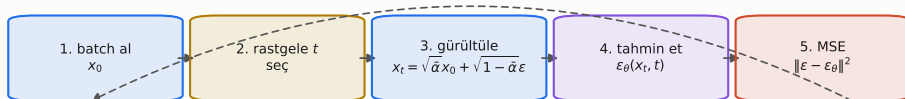
D_{KL} : **Kullback–Leibler ıraksaması** — iki dağılımın ne kadar farklı olduğunu ölçer. Burada iki dağılım da Gauss olduğundan KL kapalı formdadır ve ortalamaların farkının karesine iner. p_{θ} 'nin ortalaması uygun parametrelenirse geriye şu kalır:

Eğitim hedefi (DDPM'in basitleştirilmiş kaybı)

$$\mathcal{L}_{\text{basit}} = \mathbb{E}_{x_0, \varepsilon \sim \mathcal{N}(0, \mathbf{I}), t \sim \mathcal{U}\{1..T\}} \left[\left\| \varepsilon - \varepsilon_{\theta}(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1 - \bar{\alpha}_t}\varepsilon, t) \right\|^2 \right]$$

Bu, ortalama kare hatadan (MSE, *mean squared error*) başka bir şey değildir.

Ho ve ark., terim ağırlıklarını atmanın pratikte *daha iyi* örnek verdiğini gösterdi.



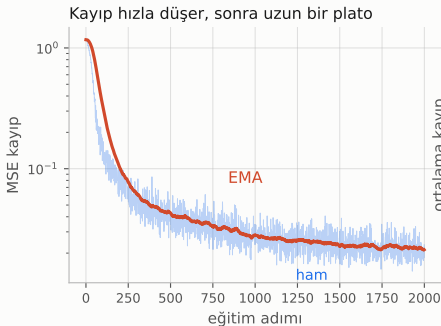
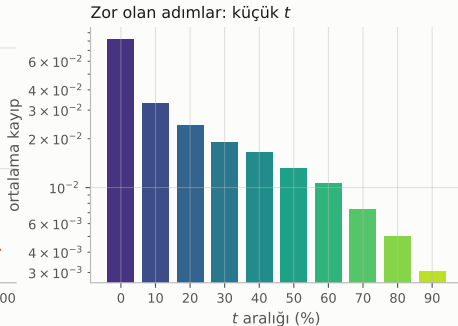
geri yayılım + optimizasyon adımı

```
for adım in range(N):  
    x0, y = batch()  
    t = randint(0, T, (B,))  
    eps = randn_like(x0)  
    xt = sqrt(ab[t])*x0 + sqrt(1-ab[t])*eps  
    kayip = mse(model(xt, t, y), eps)  
    kayip.backward(); opt.step()
```

Dikkat edilecek üç nokta

- Batch (yığın) içindeki her örnek **farklı** t alır.
- Hedef görüntü değil, **gürültüdür**.
- Etiketlerin %10'u boş etiketle değiştirilir — sebebi Bölüm 6'da.

Eğitim sırasında neye bakmalı?



Kayıp eğrisi yanıltıcıdır. İlk birkaç yüz adımda sert düşer, sonra uzun bir plato gelir — ama örnek kalitesi bu platoda gelişmeye devam eder. Koyu çizgi **EMA** (*exponential moving average*, üstel hareketli ortalama) ile yumuşatılmıştır.

Adıma göre kayıp daha bilgilendiricidir: küçük t 'de hata yüksektir, çünkü ince gürültüyü görüntünün altından ayıklamak zordur.

Model gerçekten öğreniyor mu? $\hat{x}_0$ testi

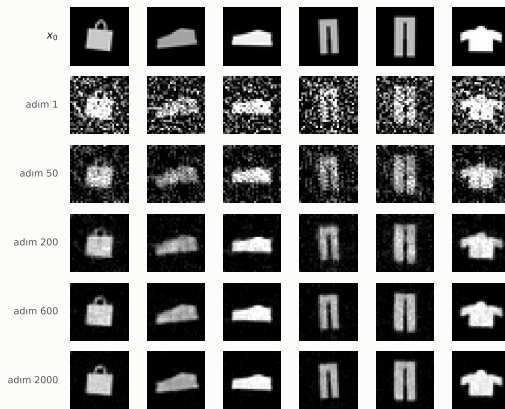
En dürüst ilerleme ölçüsü: gürültülü bir girdiden modelin tahmin ettiği temiz görüntüye bakmak.

$$\hat{x}_0 = \frac{x_t - \sqrt{1 - \bar{\alpha}_t} \varepsilon_\theta}{\sqrt{\bar{\alpha}_t}}$$

- İlk adımlarda gri bulamaç: model *ortalama kıyafeti* tahmin ediyor.
- Sonra sınıf ayrışıyor, en sonda doku geliyor.

Bu görsel, kayıp eğrisinin söylemediği şeyi söyler.

Modelin $t = 120$ girdisinden $\hat{x}_0$ tahmini



Bölüm 5 — Örnekleme

Bayes kuralıyla $q(x_{t-1} \mid x_t, x_0)$ yine Gauss çıkar:

$$q(x_{t-1} \mid x_t, x_0) = \mathcal{N}(x_{t-1}; \tilde{\mu}_t(x_t, x_0), \tilde{\beta}_t \mathbf{I}), \quad \tilde{\mu}_t = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1 - \bar{\alpha}_t}x_0 + \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t}x_t$$

x_0 'ı bilmiyoruz; modelin tahmininden $\hat{x}_0$ çıkarıp yerine koyuyoruz:

DDPM örnekleme adımı (pratikte kullanılan hâli)

$$\hat{x}_0 = \text{clip}_{[-1,1]} \left(\frac{x_t - \sqrt{1 - \bar{\alpha}_t} \varepsilon_\theta(x_t, t)}{\sqrt{\bar{\alpha}_t}} \right), \quad x_{t-1} = \tilde{\mu}_t(x_t, \hat{x}_0) + \sigma_t z, \quad z \sim \mathcal{N}(0, \mathbf{I})$$

- $\sigma_t^2 = \tilde{\beta}_t = \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t} \beta_t$; $t = 0$ 'da gürültü **eklenmez**.
- $\hat{x}_0$ yerine koyulursa bilinen kompakt biçim çıkar: $x_{t-1} = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t - \frac{\beta_t}{\sqrt{1 - \bar{\alpha}_t}} \varepsilon_\theta) + \sigma_t z$ — ama orada kırpma yapılamaz.

1. Kırpma neden şart?

Büyük t 'de $\sqrt{\alpha_t}$ çok küçüktür (bizim kurulumumuzda $\approx 0,006$). Bölme, modelin küçük hatasını ~ 30 kat büyütür. Kırpmazsanız hata adımdan adıma birikir ve süreç ıraksar — biz de test ettik: $\|x\|$ 120 kat şişti. Orijinal DDPM kodundaki `clip_denoised = True` bunun içindir.

2. Neden gürültü geri ekliyoruz?

Eklemezsek süreç her adımda koşullu ortalamaya çöker; sonuç tüm veri kümesinin bulanık ortalamasına yakınsar. Stokastiklik, çeşitliliği taşıyan şeydir.

Buradan çıkan soru: *bu gürültüyü tamamen kaldırırsak ne olur?* Cevap bir sonraki slaytta — ve bizi DDIM'e götürüyor.

DDIM: 300 adımı 25 adıma indirmek

DDIM (Denoising Diffusion Implicit Models). Song ve ark. (2021): süreçten rastgeleliği çıkarırsak ($\eta = 0$) adımları **atlayabiliriz**. İki aşama:

1. **Temiz görüntüyü tahmin et:** $\hat{x}_0 = \text{clip}\left(\frac{x_t - \sqrt{1 - \bar{\alpha}_t} \varepsilon_\theta(x_t, t)}{\sqrt{\bar{\alpha}_t}}\right)$
2. **İstediğin ara adıma yeniden gürültülendir:** $x_{t'} = \sqrt{\bar{\alpha}_{t'}} \hat{x}_0 + \sqrt{1 - \bar{\alpha}_{t'}} \varepsilon_\theta(x_t, t)$

DDPM: her adım tek tek, her adımda yeni gürültü (T ileri geçiş)

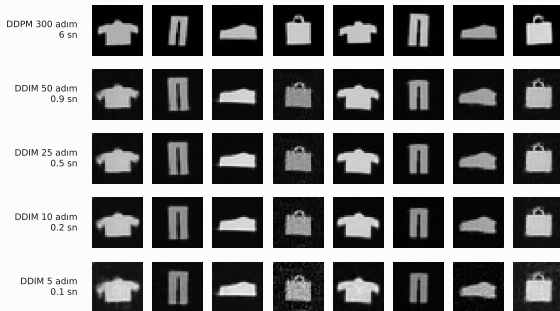


DDIM: adımları atlayarak, deterministik (≈ 25 ileri geçiş)



t' bir önceki adım olmak zorunda değildir — 299'dan doğrudan 287'ye atlanabilir. Süreç deterministiktir. Notebook: TODO-7.

Kalite – hız takası



- Aynı başlangıç gürültüsü, farklı örnekleyiciler.
- 25–50 adım, DDPM’in 300 adımına yakın kalite verir.
- Az adımda bozulmanın sebebi: her atlamada “ $\hat{x}_0$ tahminim doğru” varsayıyoruz. Model çok gürültülü girdide kötü tahmin yapar; hatayı düzeltecek fırsat kalmaz.

İleri okuma

DPM-Solver, UniPC (difüzyon denklemini daha az adımda çözen sayısal yöntemler) aynı kaliteyi 10 adımın altında verir; damıtma (distillation) 1–4 adıma indirir.

Deterministiklik ne işe yarar?

DDIM, gürültü uzayından görüntü uzayına **tersinir bir eşleme** kurar.

Bunun pratik sonuçları:

- **Tohum (seed) kontrolü:** aynı tohum $\Rightarrow$ aynı görüntü.
- **İnterpolasyon:** iki gürültü vektörü arasında yürüyerek görüntüler arasında yumuşak geçiş.
- **İnversiyon:** gerçek bir görüntüyü gürültüye geri çevirip düzenleme yapmak (görüntü düzenlemenin temeli).

η parametresi

DDIM ile DDPM arasında sürekli bir geçiş vardır:

$$\sigma_t = \eta \sqrt{\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}} \sqrt{1 - \frac{\bar{\alpha}_t}{\bar{\alpha}_{t-1}}}$$

$\eta = 0$: DDIM (deterministik).

$\eta = 1$: DDPM (tam stokastik).

Arada: çeşitlilik ile tekrarlanabilirlik arasında ayar.

Bölüm 6 — Koşullu Üretim

Ne üreteceğini nasıl söyleriz?

Altyapı zaten hazır: U-Net'e sınıf etiketini y de veriyorduk. Ama eğitimde etiketlerin %10'unu rastgele *boş* etiketiyle değiştirmiştik. Böylece tek model iki işi birden öğrendi:

$\varepsilon_{\theta}(x_t, t, y)$	"bana bir çanta göster"	(koşullu)
$\varepsilon_{\theta}(x_t, t, \emptyset)$	"ne olursa"	(koşulsuz)

Tarihçe: classifier guidance

Dhariwal & Nichol (2021) önce ayrı bir sınıflandırıcı eğitip onun gradyanını örnekleme sırasında kullandı. İşe yarıyordu ama *gürültülü görüntüler üzerinde* ayrı bir sınıflandırıcı eğitmek gerekiyordu. Ho & Salimans (2022) bu ek modelden tamamen kurtuldu.

Classifier-Free Guidance (CFG)

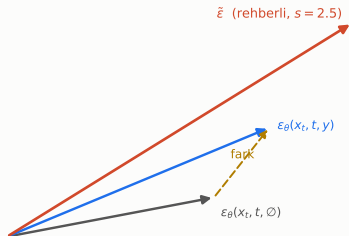
CFG = *classifier-free guidance*

(sınıflandırıcısız rehberlik). İki tahmin arasındaki **fark vektörü** "çantallığı" gösteren yöndür; o yönde abartıyoruz:

CFG

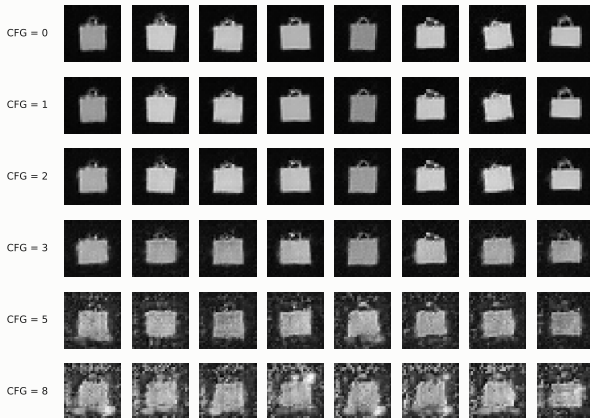
$$\tilde{\epsilon} = \epsilon_{\theta}(x_t, t, \emptyset) + s \cdot (\epsilon_{\theta}(x_t, t, y) - \epsilon_{\theta}(x_t, t, \emptyset))$$

- $s = 0$: tamamen koşulsuz $s = 1$: düz koşullu üretim
- $s = 3 \dots 8$: tipik tatlı nokta
- $s \gg 1$: aşırı doygun, bozuk görüntüler



Bu, arayüzlerdeki **CFG Scale** kaydırıcısının ta kendisidir.
Notebook: TODO-8. Maliyet: adım başına iki ileri geçiş.

Hedef sınıf: Çanta — aynı başlangıç gürültüsü



Satırlar aşağı indikçe **iki şey aynı anda** oluyor:

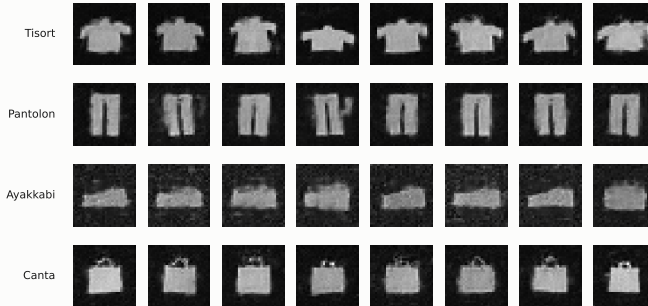
1. Görüntüler gerçekten daha çok hedef sınıfa benziyor.
2. Sekiz görüntü **birbirine benzemeye** başlıyor.

Sadakat ↔ Çeşitlilik

Üretken modellerin klasik takası. CFG, dağılımın yüksek olasılıklı bölgelerine doğru keskinleştirme yapar: örnekler tipikleşir, nadir ve yaratıcı olanlar kaybolur. Çok yüksek *s*'de kontrast da patlar.

Sınıf koşullu üretim — sonuç

Sınıf koşullu üretim (DDIM 40 adım, CFG = 3)



Her satır farklı bir hedef sınıf; DDIM 40 adım, CFG = 3. Model dört sınıfı da ayırt edilebilir biçimde üretiyor — ve bu, 0,6 milyon parametrelili, birkaç dakika eğitilmiş bir modelden alınan sonuç.

Bölüm 7 — Latent Difüzyon ve Stable Diffusion

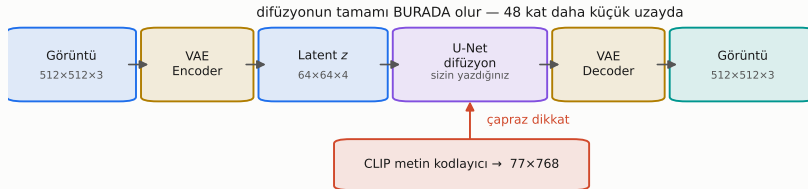
Aynı kodu 512×512 renkli görüntülere uygularsak ne olur?

	Bizim örneğimiz	512×512 RGB
Boyut sayısı	$28 \cdot 28 \cdot 1 = 784$	$512 \cdot 512 \cdot 3 = 786\,432$
Oran	$1 \times$	$\approx 1000 \times$
U-Net maliyeti (adım başına)	milisaniyeler	saniyeler
Toplam ($50 \text{ adım} \times \text{CFG}$)	anlık	dakikalar

Latent difüzyonun çözümü (Rombach ve ark., 2022)

Difüzyonu pikselde değil, **sıkıştırılmış (latent: gizli/örtük) uzayda** yap.

Stable Diffusion'ın üç parçası



Parça	Görevi	Boyut	Parametre
VAE encoder/decoder	8 kat sıkıştır / geri aç	$512^2 \times 3 \leftrightarrow 64^2 \times 4$	~84 M
CLIP metin kodlayıcı	metni vektöre çevir	"a cat" → 77×768	~123 M
U-Net	difüzyon — <i>tam sizin yazdığınız gibi</i>	$64 \times 64 \times 4$	~860 M

CLIP = Contrastive Language–Image Pre-training: metni ve görüntüyü aynı vektör uzayına yerleştiren model. Kazanç: $786\,432 \rightarrow 16\,384$ boyut, **48 kat** az; matematik birebir aynı.

VAE ve unutulmaması gereken sayı

VAE'nin çıktısı tek bir tensör değil, bir **dağılımdır** (ortalama + varyans); ondan örnek alırız.

Ayrıca ham latent'lerin standart sapması ≈ 1 değil, $\approx 4,8$ 'dir. Difüzyon matematiği $\mathcal{N}(0, 1)$ varsayımı üzerine kurulu olduğundan bir **ölçek katsayısı** uygulanır:

$$z = c \cdot \text{Enc}(x), \quad \hat{x} = \text{Dec}(z/c)$$

$$c = 0,18215$$

En sık yapılan hata

Bu tek sayı unutulursa model tamamen çöp üretir. Stable Diffusion'ı elle kuranların klasik hatasıdır.

VAE'nin sınırı

Gidiş-dönüşte ince dokular (küçük yazılar, yüz detayları) kaybolur. Stable Diffusion'ın küçük metinleri neden beceremediğinin sebeplerinden biri budur — U-Net kusursuz çalışsa bile VAE o bilgiyi taşıyamaz.

Metin koşullandırma: çapraz dikkat

Sınıf gömmesini zaman gömmesine *ekliyorduk*. Metin için bu yetmez: 77 *jetonluk* (token: metnin parçalara ayrılmış hâli) diziyi tek vektöre indirmek bilgi kaybıdır. Bunun yerine U-Net'in her bloğunda **çapraz dikkat** (cross-attention) kullanılır:

$$\text{Attn}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d}}\right) V$$

$$Q = W_Q \phi(x_t) \text{ (görüntüden)}, \quad K = W_K \tau(\text{metin}), \quad V = W_V \tau(\text{metin}) \text{ (metinden)}$$

- **Sorgu** (Q) görüntüden gelir: "bu bölgede ne olmalı?"
- **Anahtar/değer** (K, V) metinden gelir: her kelime bir cevap sunar.
- *softmax*: vektörü toplamı 1 olan olasılıklara çevirir.

Yan ürün

Bu dikkat haritaları görselleştirilebilir — hangi kelimenin görüntünün hangi bölgesini kontrol ettiğini gösterir. Prompt-to-Prompt gibi düzenleme teknikleri tam olarak bu haritaları değiştirerek çalışır.

Stable Diffusion üretim döngüsü

```
gomme = cat([kodla(""), kodla(istem)])  
latent = randn(1,4,64,64) * sigma_init  
for t in zamanlayici.timesteps:  
    girdi = cat([latent]*2)  
    cikti = unet(girdi, t, gomme)  
    e_bos, e_metin = cikti.chunk(2)  
    e = e_bos + cfg*(e_metin - e_bos)  
    latent = zamanlayici.step(e, t, latent)  
goruntu = vae.decode(latent / 0.18215)
```

Dikkat: bu döngüde *yeni* hiçbir şey yok — Bölüm 5 ve 6'da yazdığınız iki formülün latent uzaydaki hâli.
Notebook: TODO-10.

Negatif istem nedir?

Koşulsuz tahmin yerine *negatif istemin* koşullu tahminini koymaktan ibarettir:

$$\tilde{\epsilon} = \epsilon_{\text{neg}} + s(\epsilon_{\text{poz}} - \epsilon_{\text{neg}})$$

Yani “şundan uzaklaş” yönü tanımlarsınız. Bedava gelir — zaten iki geçiş yapıyorduk.

Neden SD az adımda iyi sonuç veriyor?

Kendi modelimizde 5 adımlı DDIM bozuk çıktı veriyordu; Stable Diffusion (kısaca SD) 8 adımda gayet iyi sonuç verir. Üç sebep:

1. **Model kapasitesi:** 860 M parametre, milyarlarca görüntüyle eğitilmiş. $\hat{x}_0$ tahminleri çok gürültülü girdide bile isabetli.
2. **Latent uzay daha düzgün:** VAE yüksek frekanslı detayı zaten atmış; difüzyonun öğrenmesi gereken fonksiyon daha yumuşak.
3. **Daha iyi çözücüler:** PNDM, DPM-Solver++ gibi sayısal yöntemler düz DDIM'den daha az adımda aynı yörüngeyi takip eder.

Sınırlar — derste konuşulması gereken

Eller ve parmaklar, küçük metinler, uzamsal ilişkiler ("solunda", "üstünde"), sayma ("üç kedi"), kompozisyon. Bunların çoğu CLIP metin kodlayıcısının sınırlarından ve eğitim verisindeki etiket gürültüsünden kaynaklanır — U-Net'in suçu değildir.

Toparlama

Her şey altı denklemde

1. **İleri süreç:** $x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} \varepsilon$
2. **Eğitim hedefi:** $\mathcal{L} = \mathbb{E}[\|\varepsilon - \varepsilon_\theta(x_t, t)\|^2]$
3. **Temiz görüntü tahmini:** $\hat{x}_0 = \frac{x_t - \sqrt{1 - \bar{\alpha}_t} \varepsilon_\theta}{\sqrt{\bar{\alpha}_t}}$
4. **DDPM adımı:** $x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{\sqrt{1 - \bar{\alpha}_t}} \varepsilon_\theta \right) + \sigma_t z$
5. **DDIM adımı:** $x_{t'} = \sqrt{\bar{\alpha}_{t'}} \hat{x}_0 + \sqrt{1 - \bar{\alpha}_{t'}} \varepsilon_\theta$
6. **CFG:** $\tilde{\varepsilon} = \varepsilon_\theta(\emptyset) + s(\varepsilon_\theta(y) - \varepsilon_\theta(\emptyset))$

Bu kadar

Stable Diffusion dahil, bugün kullanılan görüntü üreticilerinin tamamı bu altı denklemin üzerine kuruludur. Geri kalan her şey — mimari, ölçek, veri, çözücüler — bu çekirdeğin etrafındaki mühendisliktir.

Bölüm	TODO	Yazacağınız şey
1	TODO-1	Gürültü takvimi, α_t , $\bar{\alpha}_t$
1	TODO-2	q_sample: tek hamlede gürültülendirme
2	TODO-3	Kosinüs takvimi
3	TODO-4	Sinüzoidal zaman gömmesi
4	TODO-5	Eğitim adımı: t örnekleme + MSE hedefi
5	TODO-6	DDPM geri adımı
5	TODO-7	DDIM adımı ve adım atlama
6	TODO-8	Classifier-free guidance
7	TODO-9	VAE kodlama/çözme ve ölçek katsayısı
7	TODO-10	Stable Diffusion üretim döngüsü

Değerlendirme: “iyi örnek” nasıl ölçülür?

Neden zor?

Kayıp değeri örnek kalitesiyle zayıf ilişkilidir. Üretken modellerde *tek* bir doğru cevap yoktur — bu yüzden standart hata metrikleri işe yaramaz.

Yaygın metrikler

- **FID** (Fréchet Inception Distance): üretilen ve gerçek görüntülerin özellik dağılımları arasındaki mesafe. Düşük iyidir.
- **CLIP skoru**: metin–görüntü uyumu.
- **Precision / Recall** (kesinlik / duyarlılık): sadakat ve kapsama ayrı ölçülür.

Uyarılar

- FID örneklem sayısına çok duyarlıdır; 10k’dan az örnekle raporlanan FID güvenilmezdir.
- CFG ölçeği FID ile CLIP skorunu *ters* yönde etkiler — hangi noktayı seçtiğiniz bir tercihtir.
- Hiçbir metrik insan değerlendirmesinin yerini tutmuyor.

Kısaltmalar sözlüğü

DDPM	Denoising Diffusion Probabilistic Models — olasılıksal difüzyon modeli	EMA	Exponential Moving Average — üstel hareketli ortalama
DDIM	Denoising Diffusion Implicit Models — deterministik, adım atlayan sürüm	VAE	Variational Autoencoder — değişimsel otokodlayıcı
CFG	Classifier-Free Guidance — sınıflandırıcısız rehberlik	GAN	Generative Adversarial Network — çekişmeli üretici ağ
ELBO	Evidence Lower Bound — değişimsel alt sınır	CLIP	Contrastive Language-Image Pre-training — metin ve görüntüyü aynı uzaya koyan model
KL	Kullback-Leibler ıraksaması — dağılım farkı ölçüsü	FID	Fréchet Inception Distance — üretim kalitesi metriği
MSE	Mean Squared Error — ortalama kare hata	SD	Stable Diffusion
SNR	Signal-to-Noise Ratio — sinyal/gürültü oranı	U-Net	U biçimli kodlayıcı-çözücü ağ

Terimler sözlüğü

latent	sıkıştırılmış (gizli) uzaydaki temsil	atlama bağlantısı	encoder'dan decoder'a doğrudan bilgi (skip connection)
jeton (token)	metnin modele verilen en küçük parçası	dikkat (attention)	her konumun diğerlerine ağırlık vermesi
batch (yığın)	aynı anda işlenen örnek kümesi	çapraz dikkat	sorgu görüntüden, anahtar/değer metinden vektörü olasılıklara çevirir
epoch (devir)	veri kümesinin bir kez baştan sona işlenmesi	softmax	girdisini çıktısına ekleyen blok (residual)
tohum (seed)	rastgeleliği tekrarlanabilir kılan sayı	artık blok	değerleri geçerli aralığa sıkıştırma
istem (prompt)	modele verilen metin talimatı	kırpma (clipping)	büyük modeli az adımlıya öğretme (distillation)
negatif istem	"bundan uzak dur" diye verilen metin	damıtma	istene uygunluk ile üretim zenginliği takası
gürültü takvimi	β_t 'nin t ile değişimi (noise schedule)	sadakat/çeşitlilik	modelin yalnızca birkaç tip üretmesi
örnekleyici	geri süreci yürüten algoritma (sampler)	mod çökmesi	

- Ho, Jain, Abbeel — *Denoising Diffusion Probabilistic Models*, 2020. arXiv:2006.11239
- Song, Meng, Ermon — *Denoising Diffusion Implicit Models*, 2021. arXiv:2010.02502
- Nichol, Dhariwal — *Improved DDPM* (kosinüs takvimi), 2021. arXiv:2102.09672
- Dhariwal, Nichol — *Diffusion Models Beat GANs*, 2021. arXiv:2105.05233
- Ho, Salimans — *Classifier-Free Diffusion Guidance*, 2022. arXiv:2207.12598
- Rombach ve ark. — *High-Resolution Image Synthesis with Latent Diffusion Models*, 2022. arXiv:2112.10752
- Song ve ark. — *Score-Based Generative Modeling through SDEs*, 2021. arXiv:2011.13456
- Lilian Weng — *What are Diffusion Models?* (blog, çok iyi bir özet)

Uygulama

Bu dersin notebook'u: difuzyon_atolyesi_ogrenci.ipynb — 10 TODO, otomatik kontrol hücreleri ve canlı eğitim paneli ile.

Sorular

Uygulama notebook'u ile devam: 10 TODO sizi bekliyor.