Language Technologies Institute

Carnegie Mellon University

Multimodal Machine Learning

Lecture 11.1: Generation + Transference
Multitask and Modality Transfer

Paul Liang

** Co-lecturer: Louis-Philippe Morency.*
Original course co-developed with Tadas Baltrusaitis.
Spring 2021 and 2022 editions taught by Yonatan Bisk

Administrative Stuff

Reading Assignments are Back!

- Four main steps for the reading assignments
  - Monday 8pm: Official start of the assignment
  - Wednesday 8pm: Select your paper
  - **Friday 8pm:** Post your summary
  - **Monday 8pm:** Post your extra comments (5 posts)
- **4 papers:** multimodal multi-hop reasoning, multimodal geometric reasoning, multimodal robotics, multimodal knowledge bases.

Final Project Report (Due Sunday 12/11 at 8pm)

Main goals:

1. Produce a research paper which will motivate your research problem, describe the prior work, present your research contributions, explain the details of your experiments, and discuss your results.
2. Novel research ideas (N-1 new ideas for N students)
   - Novel algorithm
   - Novel application
3. Incorporate feedback from previous milestones
4. Compare to multimodal baselines from midterm report
   1. Did the proposed ideas solve the errors highlighted in error analysis?
   2. Broader implications of proposed ideas.

Final Project Report (Due Sunday 12/11 at 8pm)

Some suggestions:

- Proposed ideas
  - Explain how it tackles the challenges identified through error analysis
  - Formally explain the method and novelty
- Experimental setup
  - Datasets, metrics, baselines, methodology
  - Ablation studies
- Results
  - One subsection for each research question
  - The most important part is the discussion: what do the results mean, what implications they have, how should they be interpreted in the broader context?

Final Project Report (Due Sunday 12/11 at 8pm)

Some suggestions:

- Clear motivated research questions
- Clear ablation studies, revisit error analysis, add visualizations
- Not about results, but discussion
  - If it works, why does it work
  - If it doesn't idea, why did it not work and how can we fix it
- If your dataset is too large:
  - You can use a subset of your data or train for fewer epochs
  - But be consistent between experiments
- 3 students: 8 pages, 4 students: 9 pages, 5 students: 10 pages, 6 students: 11 pages

Final Project Presentations (Tuesday 12/6 and Thursday 12/8)

Main objective:

- Present your research ideas and get feedback from classmates
- Focus on only one of your new research ideas
- All students should present and answer questions
- Be sure to be on time! We have many presentations each day ☺
- All presentations are in person (no remote presentations)

Presentation length:

- 30-seconds elevator pitch
- 4-minute full presentation – all students should present

- Following each presentation, audience will be asked to share feedback

Final Project Presentations (Tuesday 12/6 and Thursday 12/8)

We will give more details about grading, presentation order, etc.

11-877 Next Semester!

Advanced Topics in MultiModal Machine Learning

11-877 • Spring 2022 • Carnegie Mellon University

Multimodal machine learning (MMML) is a vibrant multi-disciplinary research field which addresses some of the original goals of artificial intelligence by integrating and modeling multiple communicative modalities, including language, vision, and acoustic. This research field brings some unique challenges for multimodal researchers given the heterogeneity of the data and the contingency often found between modalities. This course is designed to be a graduate-level course covering recent research papers in multimodal machine learning, including technical challenges with representation, alignment, reasoning, generation, co-learning and quantifications. The main goal of the course is to increase critical thinking skills, knowledge of recent technical achievements, and understanding of future research directions.

- **Time:** Friday 10:10-11:30 am
- **Location:** Virtual for the first 2 weeks (find zoom link in piazza), GHC 5222 thereafter
- **Discussion and Q&A:** Piazza
- **Assignment submissions:** Canvas (for registered students only)
- **Contact:** Students should ask all course-related questions on Piazza, where you will also find announcements.

Instructor Louis-Philippe Morency
Email: morency@cs.cmu.edu

Instructor Amir Zadeh
Email: abagherz@cs.cmu.edu

Instructor Paul Liang
Email: pliang@cs.cmu.edu

1/28 Week 2: **Cross-modal interactions** [synopsis]

- What are the different ways in which modalities can interact with each other in multimodal tasks? Can we formalize a taxonomy of such cross-modal interactions, which will enable us to compare and contrast them more precisely?
- What are the design decisions (aka inductive biases) that can be used when modeling these cross-modal interactions in machine learning models?
- What are the advantages and drawbacks of designing models to capture each type of cross-modal interaction? Consider not just prediction performance, but tradeoffs in time/space complexity, interpretability, etc.
- Given an arbitrary dataset and prediction task, how can we systematically decide what type of cross-modal interactions exist, and how can that inform our modeling decisions?
- Given trained multimodal models, how can we understand or visualize the nature of cross-modal interactions?

- Does my multimodal model learn cross-modal interactions? It's harder to tell than you might think!
- What Does BERT with Vision Look At?
- Multiplicative Interactions and Where to Find Them
- Cooperative Learning for Multi-view Analysis
- Vision-and-Language or Vision-for-Language? On Cross-Modal Influence in Multimodal Transformers
- Seeing past words: Testing the cross-modal capabilities of pretrained V&L models on counting tasks

2/4 Week 3: **Multimodal co-learning** [synopsis]

- What are the types of cross-modal interactions involved to enable such co-learning scenarios where multimodal training ends up generalizing to unimodal testing?
- What are some design decisions (inductive bias) that could be made to promote transfer of information from one modality to another?
- How do we ensure that during co-learning, only useful information is transferred, and not some undesirable bias? This may become a bigger issue in low-resource settings.
- How can we know if co-learning has succeeded? Or failed? What approaches could we develop to visualize and probe the success of co-learning?
- How can we formally, empirically, or intuitively measure the additional information provided by auxiliary modality? How can we design controlled experiments to test these hypotheses?
- What are the advantages and drawbacks of information transfer during co-learning? Consider not just prediction performance, but also tradeoffs with complexity, interpretability, fairness, etc.

- Multimodal Prototypical Networks for Few-shot Learning
- SMIL: Multimodal Learning with Severely Missing Modality
- Multimodal Co-learning: Challenges, Applications with Datasets, Recent Advances and Future Directions
- Vokenization: Improving Language Understanding with Contextualized, Visual-Grounded Supervision
- What Makes Multi-modal Learning Better than Single (Provably)
- Found in Translation: Learning Robust Joint Representations by Cyclic Translations Between Modalities
- Zero-Shot Learning Through Cross-Modal Transfer
- 12-in-1: Multi-Task Vision and Language Representation Learning
- A Survey of Reinforcement Learning Informed by Natural Language

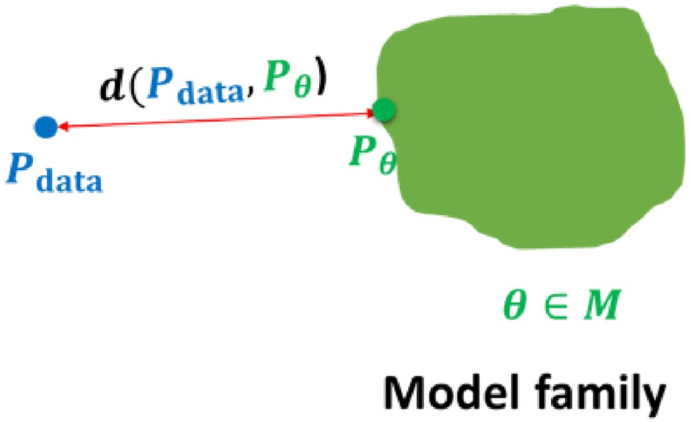
https://cmu-multicomp-lab.github.io/adv-mmml-course/spring2022/

Generative Adversarial Networks

Beyond likelihood-based learning:

- Difficulty in evaluating and optimizing p(x) in high-dimensions
- High p(x) might not correspond to realistic samples

$\mathbf{x}_i \sim P_{\text{data}}$
$i = 1, 2, \ldots, n$



Generative Adversarial Networks

Towards likelihood-free learning

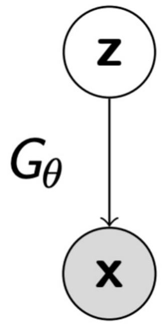
vs.

$S_1 = \{\mathbf{x} \sim P\}$ $S_2 = \{\mathbf{x} \sim Q\}$

Given a finite set of samples from two distributions, how can we tell if these samples are from the same distribution? (i.e. P = Q?)

Generative Adversarial Networks

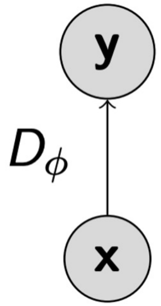
- A 2 player minimax game between a **generator** and a **discriminator**



- **Generator**: a directed latent variable model from z to x

Generative Adversarial Networks

- A 2 player minimax game between a **generator** and a **discriminator**



- **Discriminator**: any function (e.g. neural network) that tries to distinguish ‘real’ samples from the datasets from ‘fake’ samples generated by the model

GAN Training

- Training objective for **discriminator:**

$$\max_D V(G, D) = E_{\mathbf{x}\sim p_{\text{data}}}[\log D(\mathbf{x})] + E_{\mathbf{x}\sim p_G}[\log(1 - D(\mathbf{x}))]$$

- For a fixed generator G, the discriminator performs binary classification between true samples (assign label 1) vs generated samples (assign label 0)

- Training objective for **generator:**

$$\min_G V(G, D) = E_{\mathbf{x}\sim p_{\text{data}}}[\log D(\mathbf{x})] + E_{\mathbf{x}\sim p_G}[\log(1 - D(\mathbf{x}))]$$

$$= E_{\mathbf{x}\sim p_G}[\log(1 - D(\mathbf{x}))]$$

- Generator attempts to fool the discriminator to assign high likelihood to generated samples

GAN Training

- Sample minibatch of m training points $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(m)}$ from $\mathcal{D}$
- Sample minibatch of m noise vectors $\mathbf{z}^{(1)}, \mathbf{z}^{(2)}, \ldots, \mathbf{z}^{(m)}$ from p_z
- Update the generator parameters θ by stochastic gradient **descent**

$$\nabla_\theta V(G_\theta, D_\phi) = \frac{1}{m}\nabla_\theta \sum_{i=1}^{m} \log(1 - D_\phi(G_\theta(\mathbf{z}^{(i)})))$$

- Update the discriminator parameters ϕ by stochastic gradient **ascent**

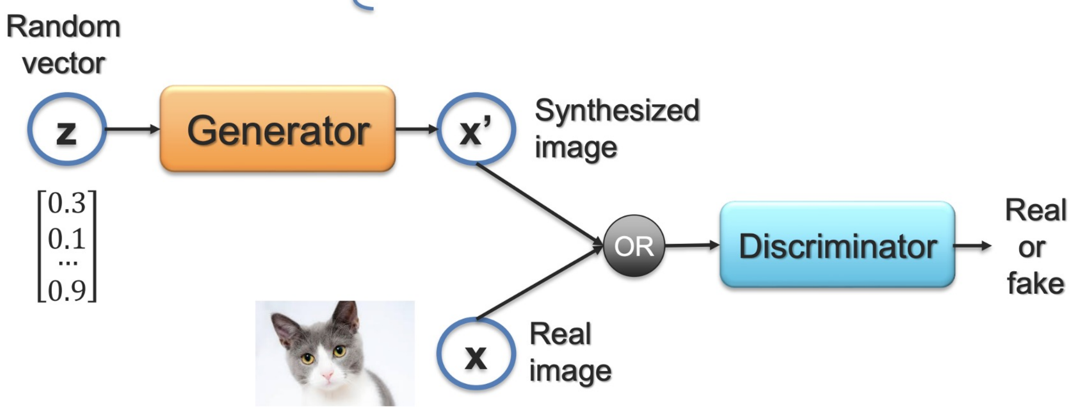
$$\nabla_\phi V(G_\theta, D_\phi) = \frac{1}{m}\nabla_\phi \sum_{i=1}^{m} [\log D_\phi(\mathbf{x}^{(i)}) + \log(1 - D_\phi(G_\theta(\mathbf{z}^{(i)})))]$$

- Repeat for fixed number of epochs

$$\max_{\mathcal{D}} \min_{\mathcal{G}} V(\mathcal{G}, \mathcal{D})$$

Optimization:

1. Fix generator, and update discriminator
2. Fix discriminator, and update generator



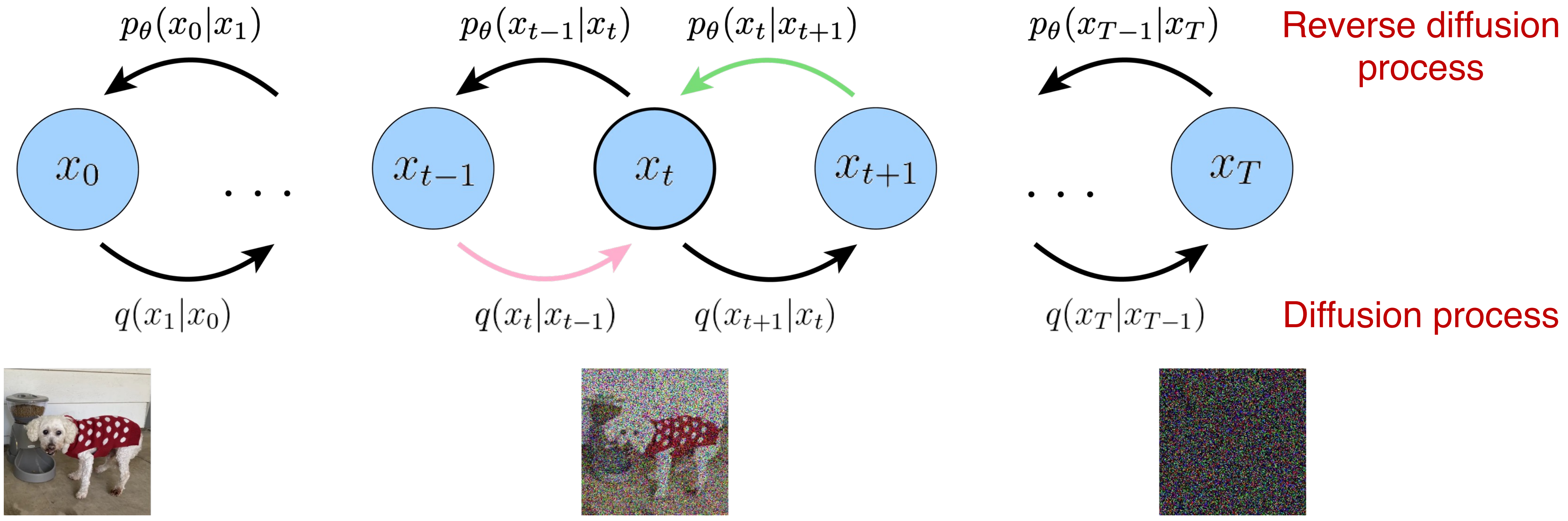
[Slides from Ermon and Grover]

Summary: Generative Models

Likelihood-based		
1. VAEs – approximate inference via evidence lower bound	Fast & easy to train	Lower generation quality
2. Autoregressive models – exact inference via chain rule	Easy to train, exact likelihood	Slow to sample from
3. Flows – exact inference via invertible transformations	Easy to train, exact likelihood	Constrained architecture
Likelihood-free		
1. GANs – discriminative real vs generated samples	High generation quality	Hard to train, can't get features

Diffusion Models

Generative modeling via denoising



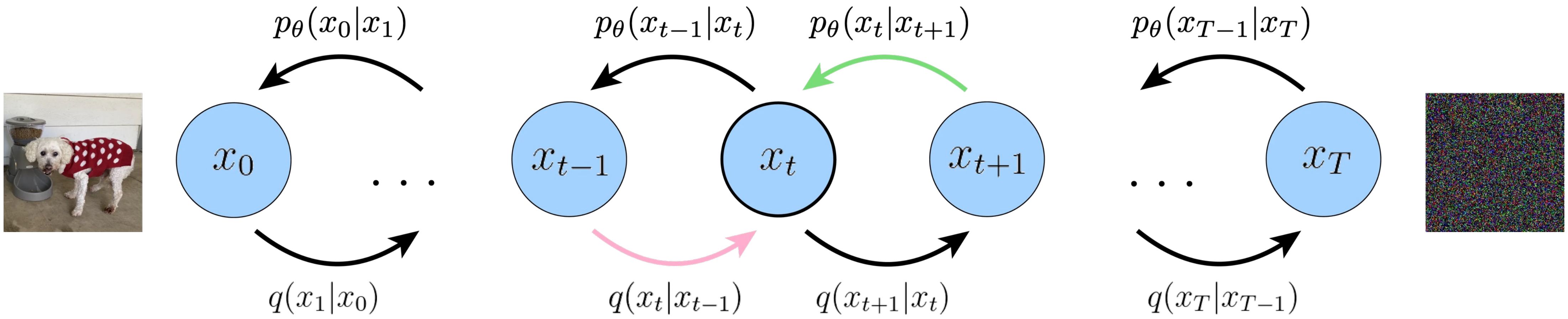
Encoding via adding noise: $q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{\alpha_t}\boldsymbol{x}_{t-1}, (1-\alpha_t)\mathbf{I})$ Noise parameters

Decoding via denoising: $p(\boldsymbol{x}_{0:T}) = p(\boldsymbol{x}_T)\prod_{t=1}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t) \quad \text{where } p(\boldsymbol{x}_T) = \mathcal{N}(\boldsymbol{x}_T; \mathbf{0}, \mathbf{I})$

[Tutorial by Calvin Luo and Yang Song]

Diffusion Models

Generative modeling via denoising



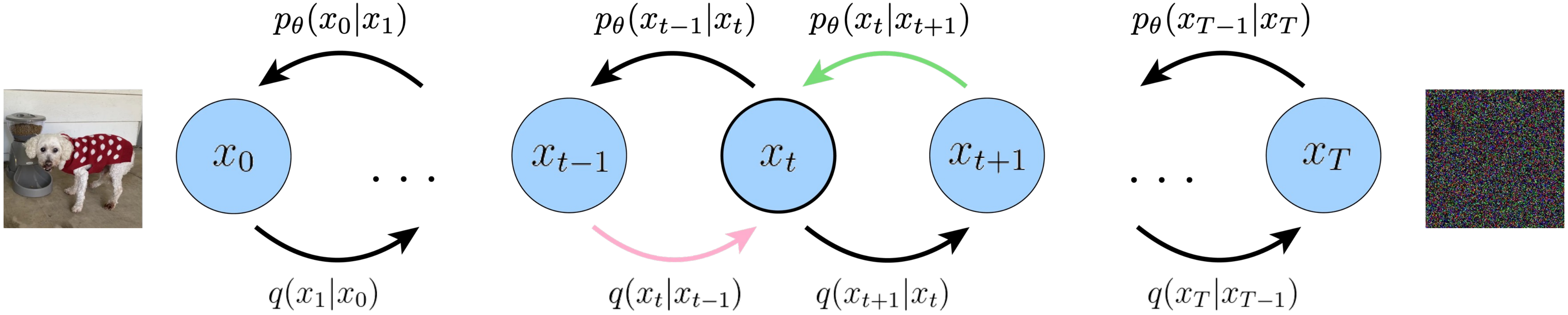
Similar to variational autoencoder, but:

1. The latent dimension is exactly equal to the data dimension.
2. Encoder q is not learned, but pre-defined as a Gaussian distribution centered around the output of previous timestep.
3. Gaussian parameters of latent encoders vary over time such that distribution of final latent is a standard Gaussian.

[Tutorial by Calvin Luo and Yang Song]

Learning Diffusion Models

Key idea: use variational inference

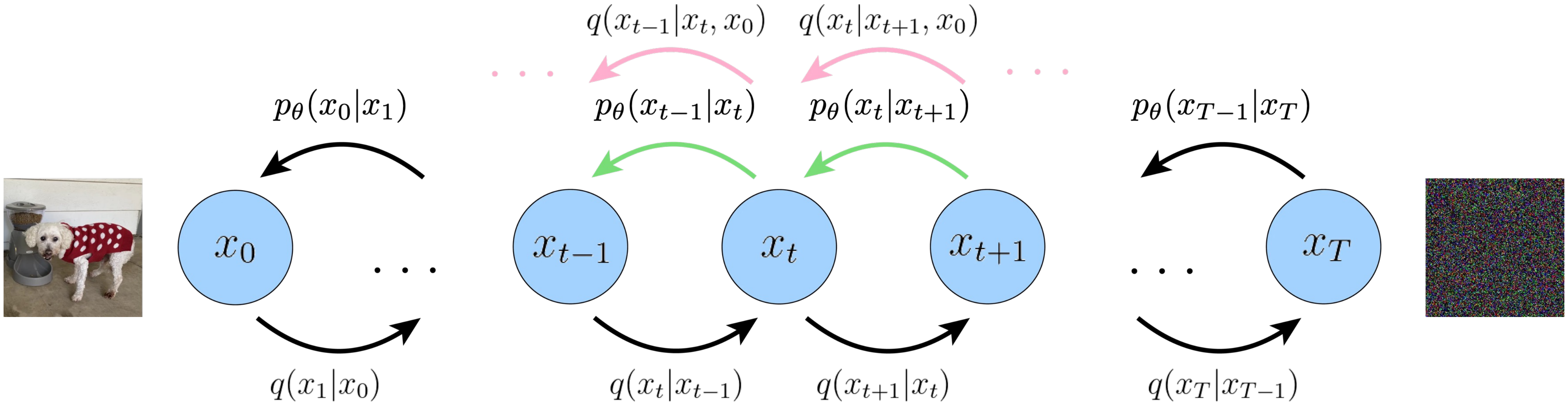


$$
\begin{aligned}
\log p(\boldsymbol{x}) &\geq \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)}\left[\log \frac{p(\boldsymbol{x}_{0:T})}{q(\boldsymbol{x}_{1:T} \mid \boldsymbol{x}_0)}\right] \\
&= \underbrace{\mathbb{E}_{q(\boldsymbol{x}_1|\boldsymbol{x}_0)}\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0 \mid \boldsymbol{x}_1)\right]}_{\text{reconstruction term}} - \underbrace{\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{x}_T \mid \boldsymbol{x}_0) \,||\, p(\boldsymbol{x}_T))}_{\text{prior matching term}} \\
&\quad - \sum_{t=2}^{T} \underbrace{\mathbb{E}_{q(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) \,||\, p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t))\right]}_{\text{denoising matching term}}
\end{aligned}
$$

[Tutorial by Calvin Luo and Yang Song]

Learning Diffusion Models

Key idea: use variational inference



$$-\sum_{t=2}^{T} \underbrace{\mathbb{E}_{q(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) \mid\mid p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t))\right]}_{\text{denoising matching term}}$$

[Tutorial by Calvin Luo and Yang Song]

Learning Diffusion Models

$$-\sum_{t=2}^{T} \underbrace{\mathbb{E}_{q(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) \mid\mid p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t))\right]}_{\text{denoising matching term}}$$

$$q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_0) = q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{\alpha_t}\boldsymbol{x}_{t-1}, (1-\alpha_t)\mathbf{I})$$

Reparameterization as adding noise:

$$\boldsymbol{x}_t = \sqrt{\alpha_t}\boldsymbol{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon} \quad \text{with } \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}, \mathbf{I})$$

$$\boldsymbol{x}_{t-1} = \sqrt{\alpha_{t-1}}\boldsymbol{x}_{t-2} + \sqrt{1-\alpha_{t-1}}\boldsymbol{\epsilon} \quad \text{with } \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}, \mathbf{I})$$

$$\boldsymbol{x}_t \sim \mathcal{N}(\boldsymbol{x}_t; \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0, (1-\bar{\alpha}_t)\,\mathbf{I})$$

$$q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) = \frac{q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_0)q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_0)}{q(\boldsymbol{x}_t \mid \boldsymbol{x}_0)}$$

Essentially this is proportional to a Gaussian $\mathcal{N}\left(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q\left(t\right)\right)$

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t) = \mathcal{N}\left(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_q\left(t\right)\right)$$

Also parameterize this as a Gaussian model

$$\begin{aligned} &\arg\min_{\boldsymbol{\theta}} \mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) \mid\mid p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)) \\ &= \arg\min_{\boldsymbol{\theta}} \mathcal{D}_{\mathrm{KL}}\left(\mathcal{N}\left(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q\left(t\right)\right) \mid\mid \mathcal{N}\left(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_q\left(t\right)\right)\right) \\ &= \arg\min_{\boldsymbol{\theta}} \frac{1}{2\sigma_q^2(t)}\left[\left\|\boldsymbol{\mu}_{\boldsymbol{\theta}} - \boldsymbol{\mu}_q\right\|_2^2\right] \end{aligned}$$

[Tutorial by Calvin Luo and Yang Song]

Learning Diffusion Models

$$-\sum_{t=2}^{T} \underbrace{\mathbb{E}_{q(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) \,||\, p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t))\right]}_{\text{denoising matching term}}$$

$$q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) = \mathcal{N}\left(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q(t)\right) \qquad \boldsymbol{\mu}_q(\boldsymbol{x}_t, \boldsymbol{x}_0) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\boldsymbol{x}_0}{1-\bar{\alpha}_t}$$

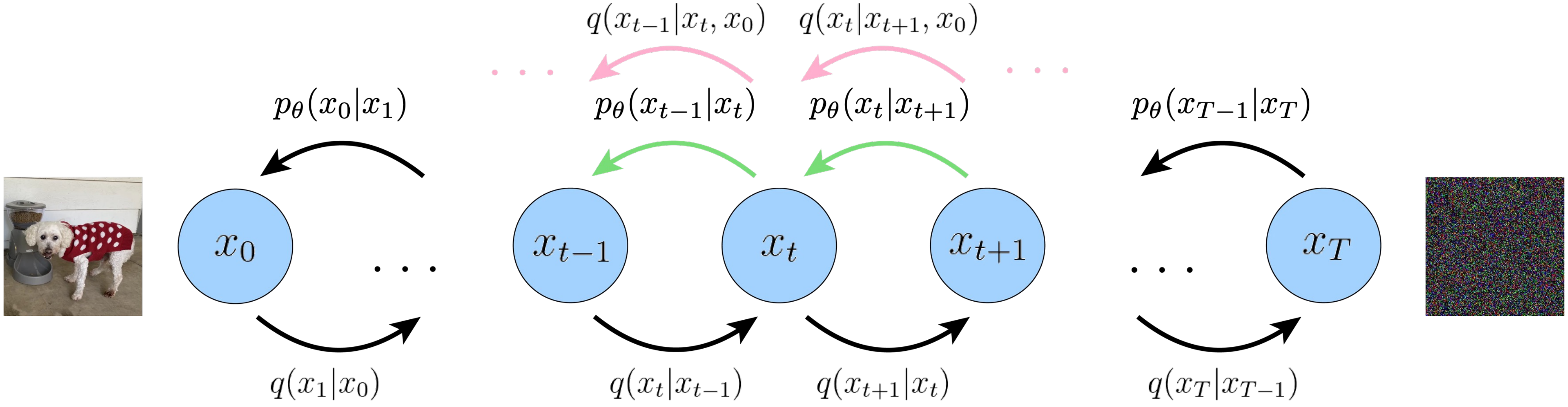
$$p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t) = \mathcal{N}\left(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_q(t)\right) \qquad \boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)}{1-\bar{\alpha}_t}$$

Neural network to predicts perfect image x_0 from noisy image x_t at time t.

$$\begin{aligned}
&\arg\min_{\boldsymbol{\theta}} \mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) \,||\, p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)) \\
&= \arg\min_{\boldsymbol{\theta}} \mathcal{D}_{\mathrm{KL}}\left(\mathcal{N}\left(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q(t)\right) \,||\, \mathcal{N}\left(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_q(t)\right)\right) \\
&= \arg\min_{\boldsymbol{\theta}} \frac{1}{2\sigma_q^2(t)} \frac{\bar{\alpha}_{t-1}(1-\alpha_t)^2}{(1-\bar{\alpha}_t)^2}\left[\left\|\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \boldsymbol{x}_0\right\|_2^2\right]
\end{aligned}$$

[Tutorial by Calvin Luo and Yang Song]

Learning Diffusion Models



$$\log p(\boldsymbol{x}) \geq \underbrace{\mathbb{E}_{q(\boldsymbol{x}_1|\boldsymbol{x}_0)}\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0 \mid \boldsymbol{x}_1)\right]}_{\text{reconstruction term}} - \underbrace{\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{x}_T \mid \boldsymbol{x}_0) \,||\, p(\boldsymbol{x}_T))}_{\text{prior matching term}} - \sum_{t=2}^{T} \underbrace{\mathbb{E}_{q(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) \,||\, p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t))\right]}_{\text{denoising matching term}}$$

$$\arg\min_{\boldsymbol{\theta}} \sum_{t=2}^{T} \mathbb{E}_{q(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) \,||\, p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t))\right]$$

$$= \arg\min_{\boldsymbol{\theta}} \mathbb{E}_{t\sim U\{2,T\}}\left[\mathbb{E}_{q(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\frac{1}{2\sigma_q^2(t)} \frac{\bar{\alpha}_{t-1}(1-\alpha_t)^2}{(1-\bar{\alpha}_t)^2}\left[\|\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \boldsymbol{x}_0\|_2^2\right]\right]\right]$$

[Tutorial by Calvin Luo and Yang Song]

Learning Noise Parameters

$$q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_0) = q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{\alpha_t}\boldsymbol{x}_{t-1}, (1-\alpha_t)\mathbf{I})$$

Reparameterization:

$$\boldsymbol{x}_t = \sqrt{\alpha_t}\boldsymbol{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon} \quad \text{with } \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}, \mathbf{I})$$

$$\boldsymbol{x}_{t-1} = \sqrt{\alpha_{t-1}}\boldsymbol{x}_{t-2} + \sqrt{1-\alpha_{t-1}}\boldsymbol{\epsilon} \quad \text{with } \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}, \mathbf{I})$$

$$\boldsymbol{x}_t \sim \mathcal{N}(\boldsymbol{x}_t; \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0, (1-\bar{\alpha}_t)\,\mathbf{I})$$

$$\bar{\alpha}_t = \prod_i \alpha_i$$

Choose $\bar{\alpha}_1 > \cdots > \bar{\alpha}_T$

i.e., add smaller noise at the beginning of the diffusion process and gradually increase noise when the samples get noisier.

[Tutorial by Calvin Luo and Yang Song]

Diffusion Models Interpretations

3 related interpretations:

1. Learning a model to predict original image x_0 from noisy image x_t at timestep t.

$$q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) = \mathcal{N}\left(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q\left(t\right)\right) \qquad \boldsymbol{\mu}_q(\boldsymbol{x}_t, \boldsymbol{x}_0) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\boldsymbol{x}_0}{1-\bar{\alpha}_t}$$

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t) = \mathcal{N}\left(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_q\left(t\right)\right) \qquad \boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)}{1-\bar{\alpha}_t}$$

2. Learning a model to predict the noise ϵ_t added at timestep t.

$$q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_0) = q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{\alpha_t}\boldsymbol{x}_{t-1}, (1-\alpha_t)\mathbf{I})$$

$$\boldsymbol{x}_t = \sqrt{\alpha_t}\boldsymbol{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon} \quad \text{with } \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}, \mathbf{I})$$

$$\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_0$$

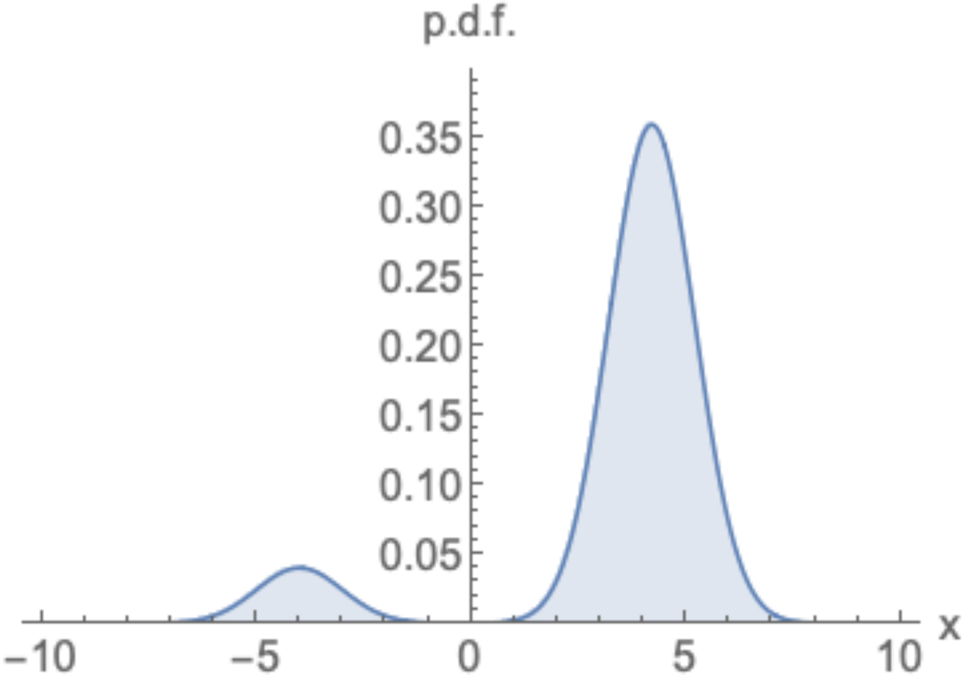
$$\begin{aligned}\boldsymbol{\mu}_q(\boldsymbol{x}_t, \boldsymbol{x}_0) &= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\boldsymbol{x}_0}{1-\bar{\alpha}_t} \\ &= \frac{1}{\sqrt{\alpha_t}}\boldsymbol{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}\sqrt{\alpha_t}}\boldsymbol{\epsilon}_0\end{aligned} \qquad \boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\boldsymbol{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}\sqrt{\alpha_t}}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$$

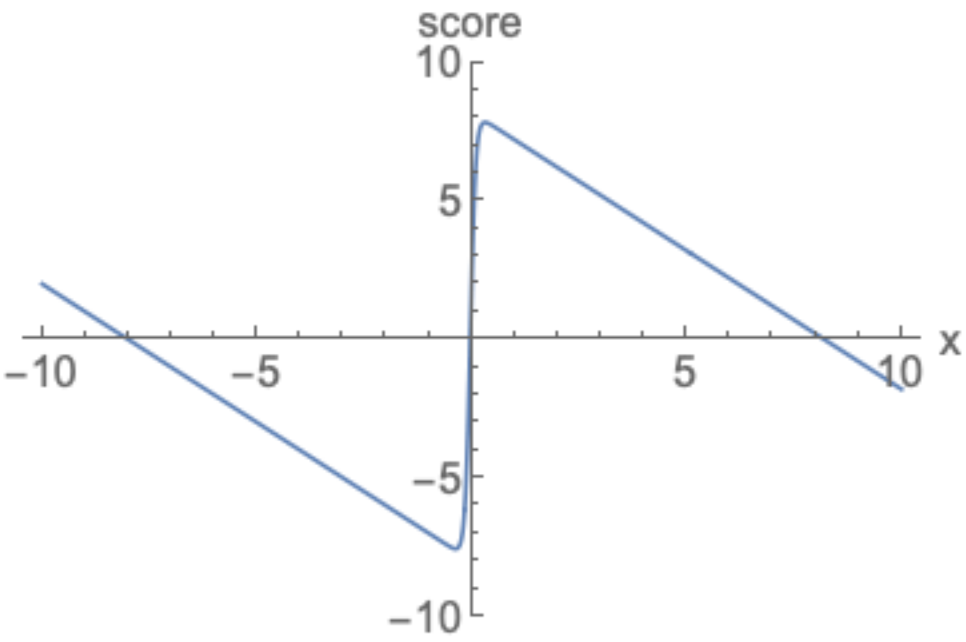
Diffusion Models and Score-based Models

3 related interpretations:

3. Learning a model to predict the score function – these are good because they don't need to be normalized!

$$\mathbb{E}_{p(\boldsymbol{x})}\left[\|\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x}) - \nabla \log p(\boldsymbol{x})\|_2^2\right]$$





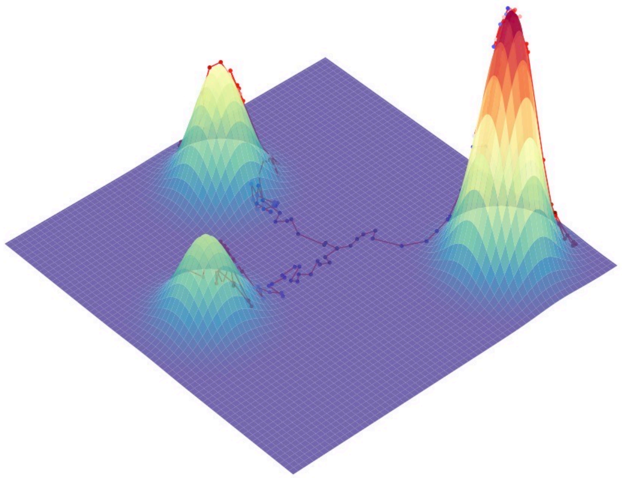
[Tutorial by Yang Song]

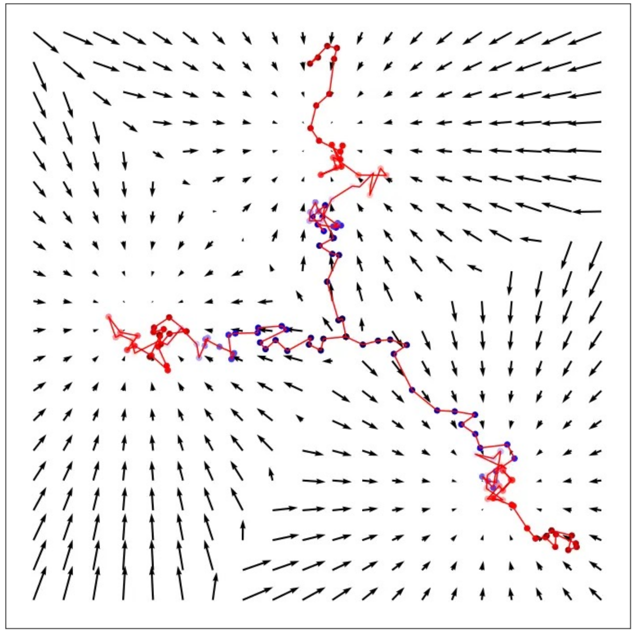
Diffusion Models and Score-based Models

3 related interpretations:

3. Learning a model to predict the score function: score matching

$$\mathbb{E}_{p(\boldsymbol{x})}\left[\|\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x})-\nabla \log p(\boldsymbol{x})\|_2^2\right]$$



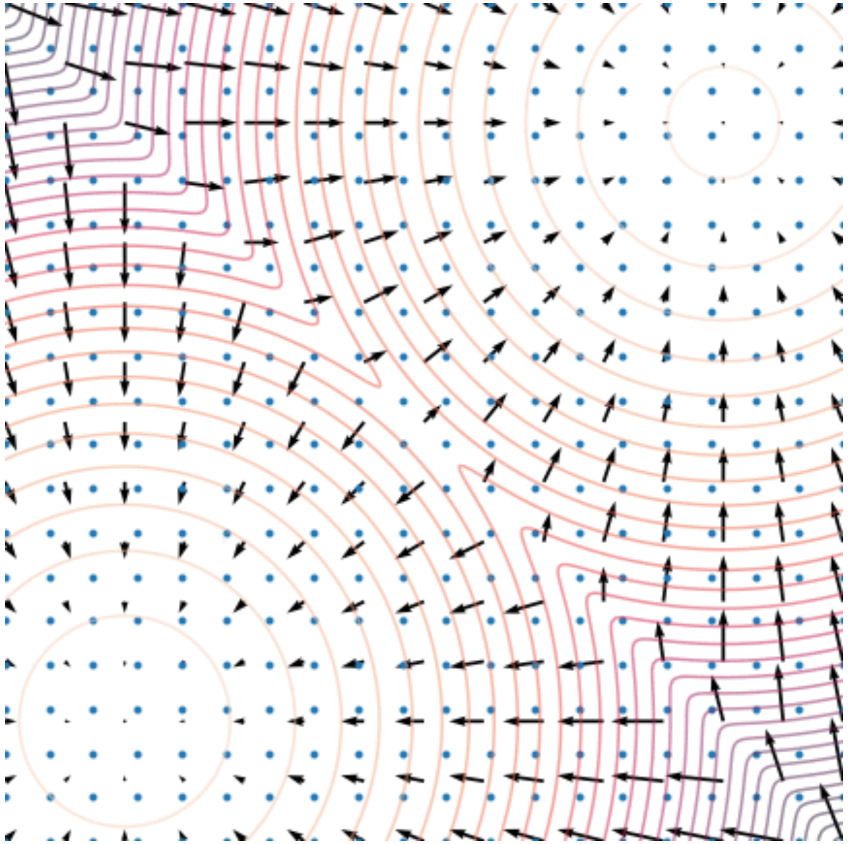


[Tutorial by Yang Song]

Diffusion Models and Score-based Models

Using the score function to sampling with Langevin dynamics

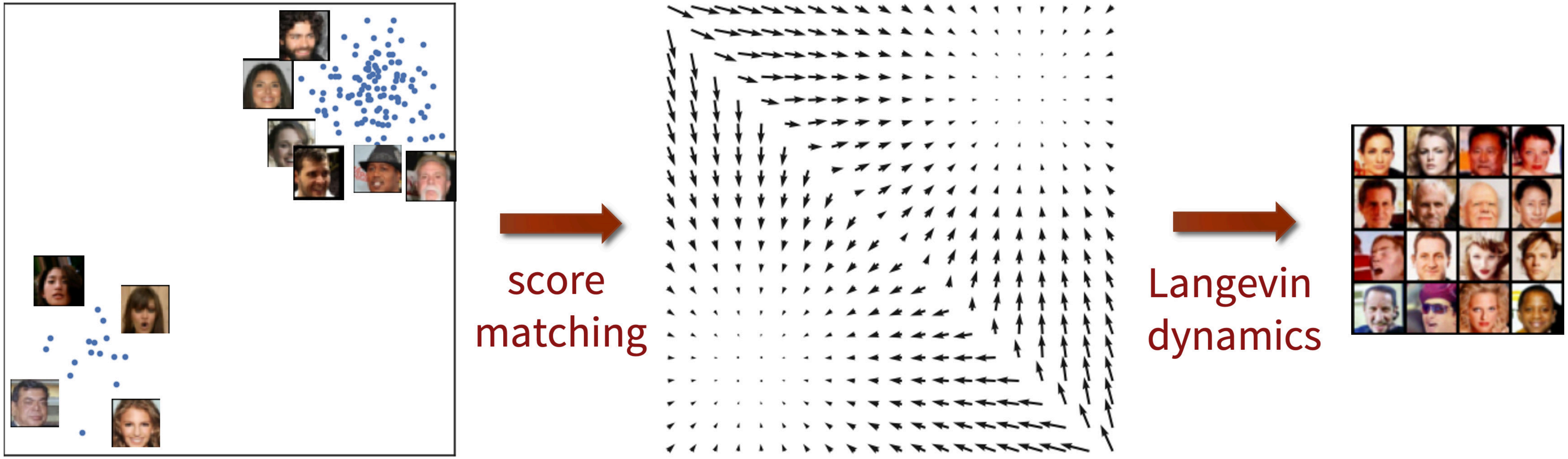
$$\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{x}_i + c\nabla \log p(\boldsymbol{x}_i) + \sqrt{2c}\boldsymbol{\epsilon}, \quad i = 0, 1, \dots, K$$



[Tutorial by Yang Song]

Diffusion Models and Score-based Models

Score-based models



Data samples

$$\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\} \overset{\text{i.i.d.}}{\sim} p(\mathbf{x})$$

Scores

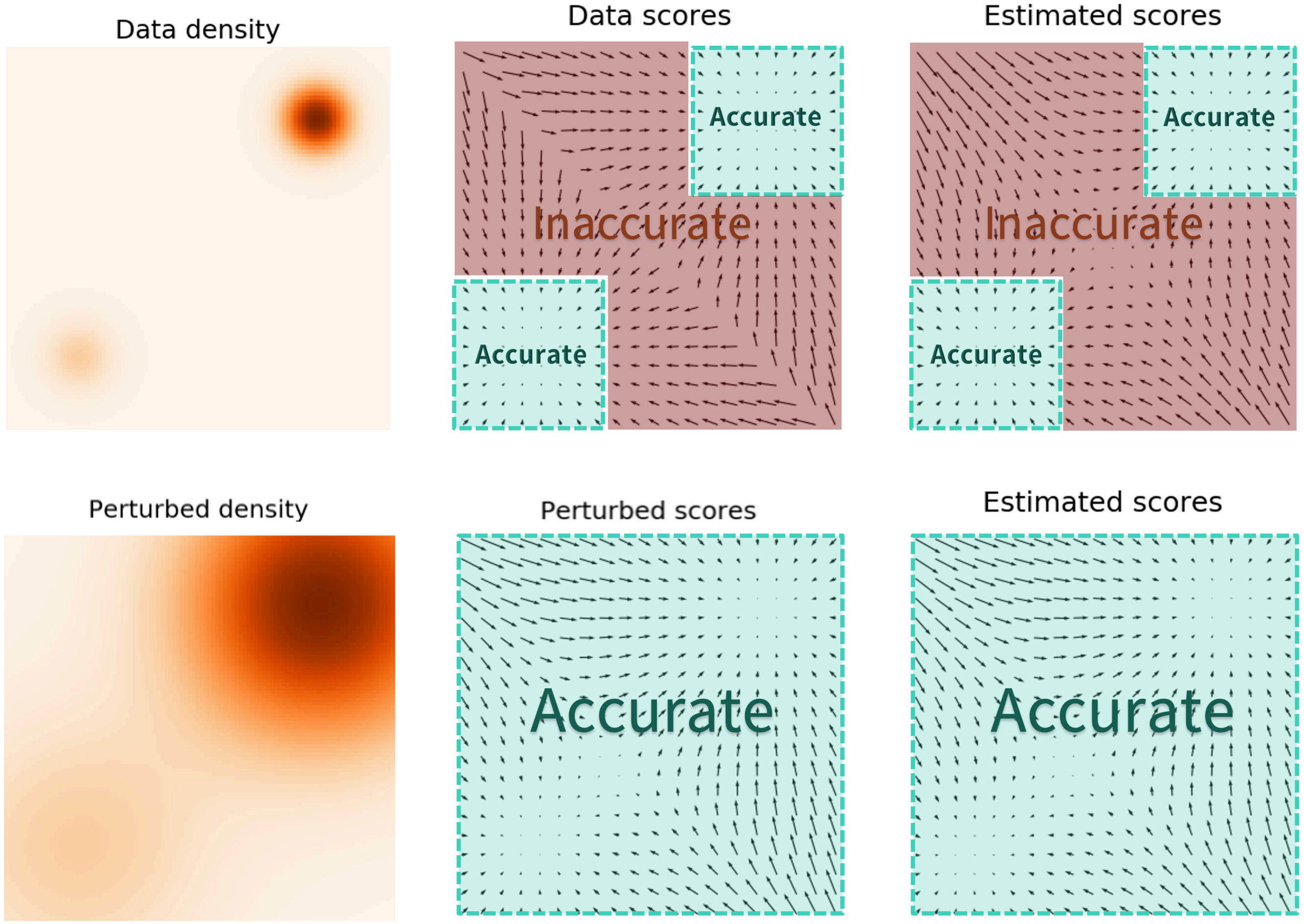
$$\mathbf{s}_\theta(\mathbf{x}) \approx \nabla_{\mathbf{x}} \log p(\mathbf{x})$$

New samples

[Tutorial by Yang Song]

Diffusion Models and Score-based Models

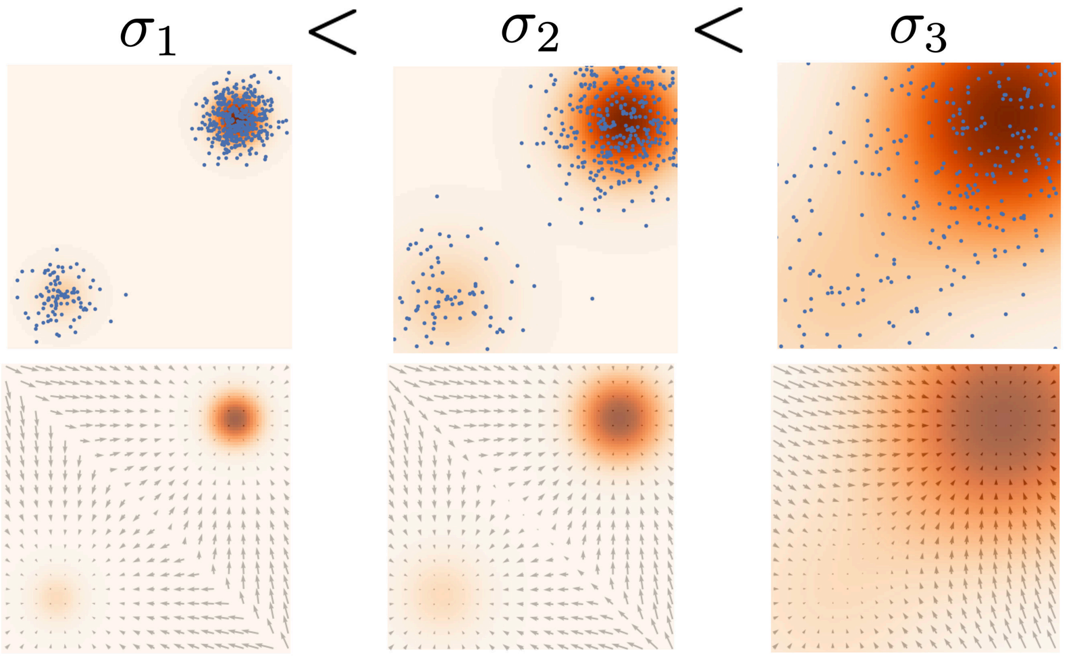
Difficulties with score-based models



[Tutorial by Yang Song]

Diffusion Models and Score-based Models

How much noise to add?

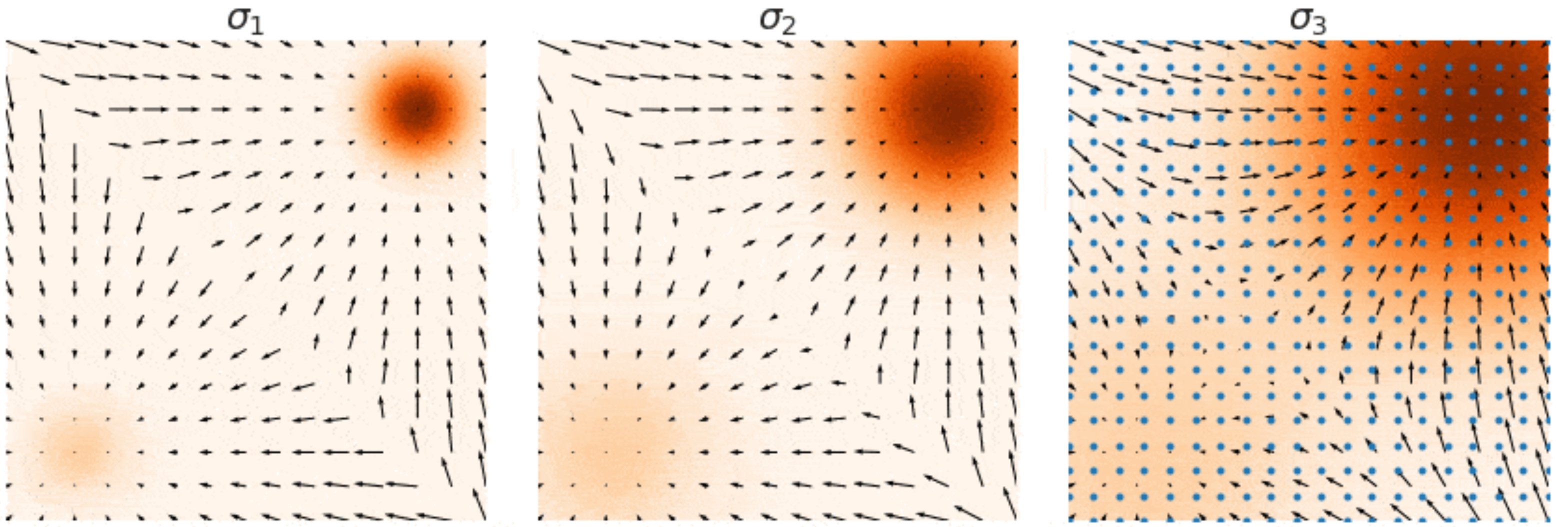


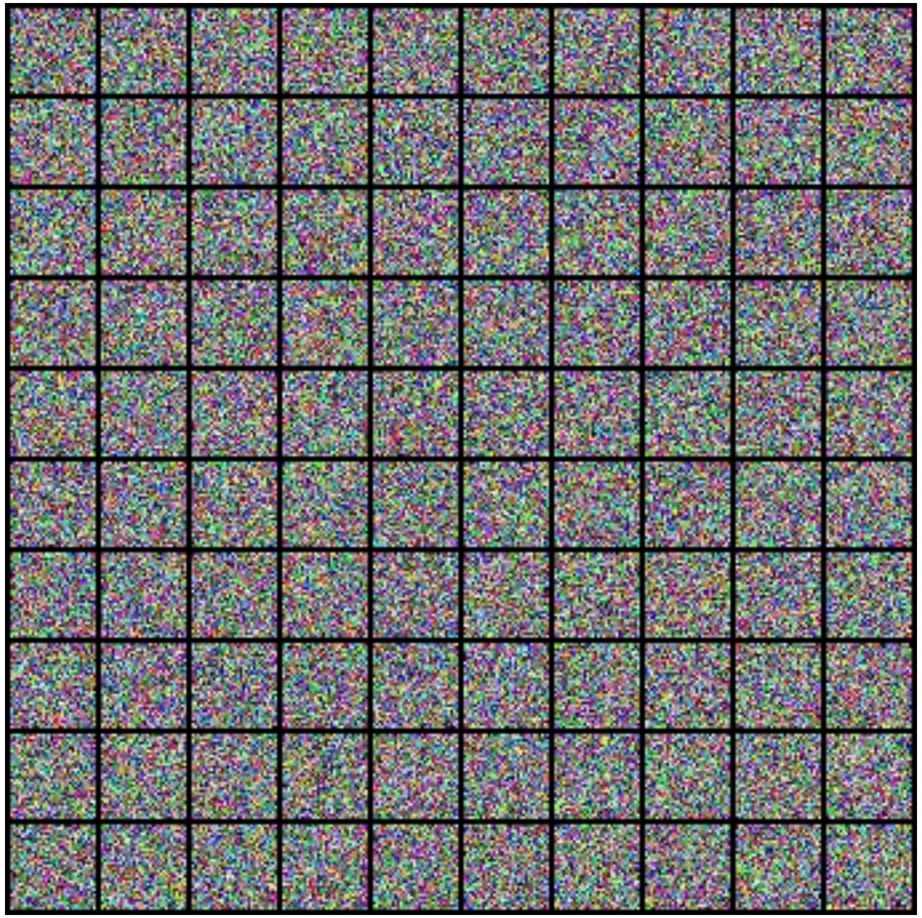
Add increasing scales of Gaussian noise and estimate score functions for all of them!

$$\mathbf{s}_\theta(\mathbf{x}, i) \approx \nabla_\mathbf{x} \log p_{\sigma_i}(\mathbf{x}) \qquad \sum_{i=1}^{L} \lambda(i) \mathbb{E}_{p_{\sigma_i}(\mathbf{x})}[\|\nabla_\mathbf{x} \log p_{\sigma_i}(\mathbf{x}) - \mathbf{s}_\theta(\mathbf{x}, i)\|_2^2],$$

Diffusion Models and Score-based Models

Diffusion Models: score-based methods with annealed Langevin sampling



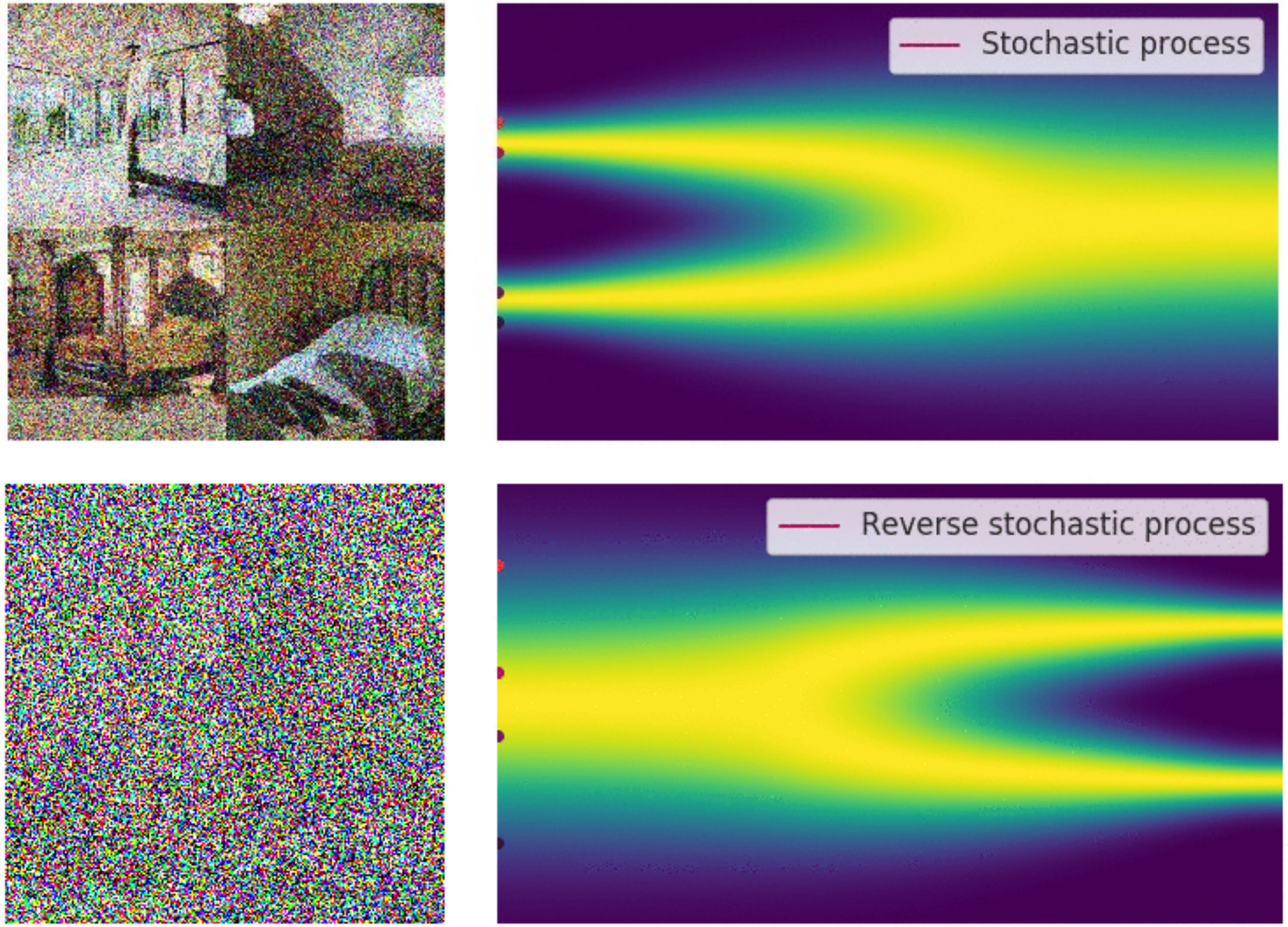


[Tutorial by Yang Song]

Diffusion Models as Differential Equations

From discrete diffusion process to continuous diffusion process

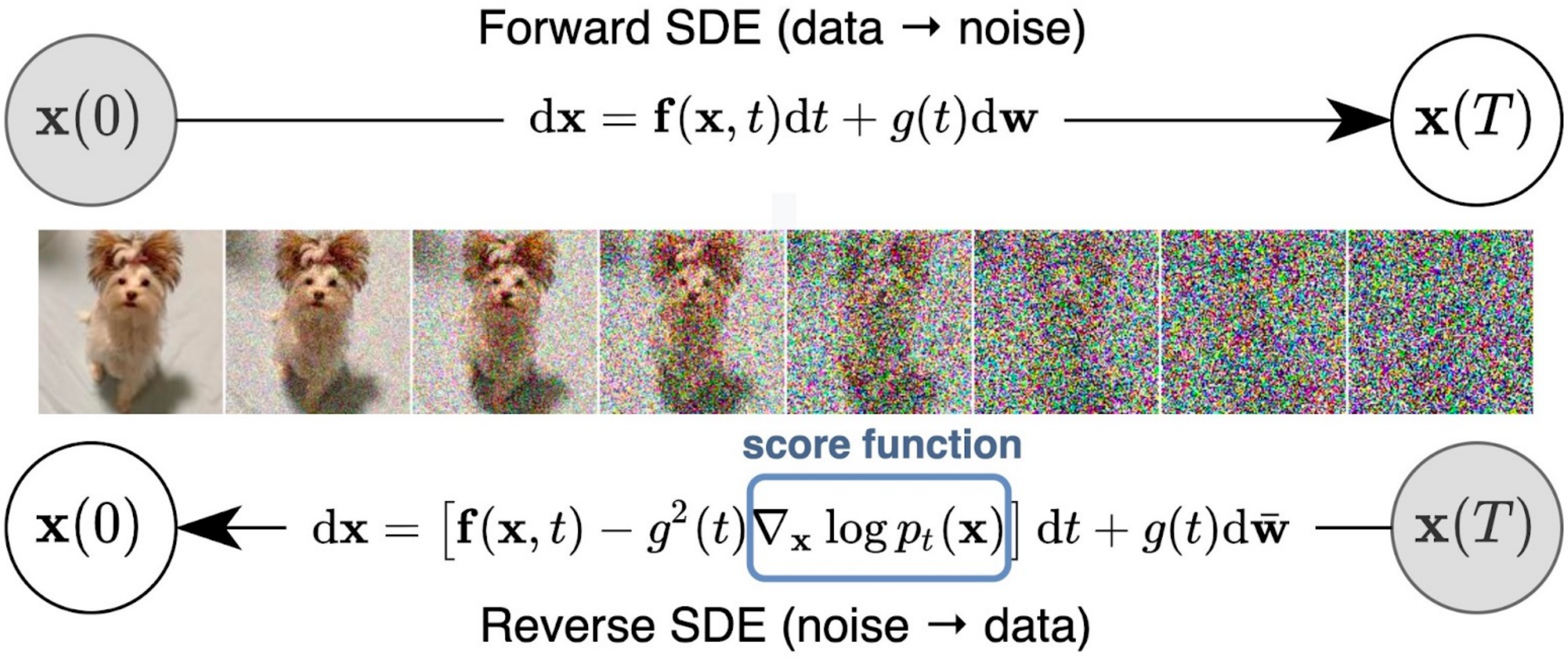
- Higher quality samples
- Exact log-likelihood
- Controllable generation



[Tutorial by Yang Song]

Diffusion Models as Differential Equations

From discrete diffusion process to continuous diffusion process

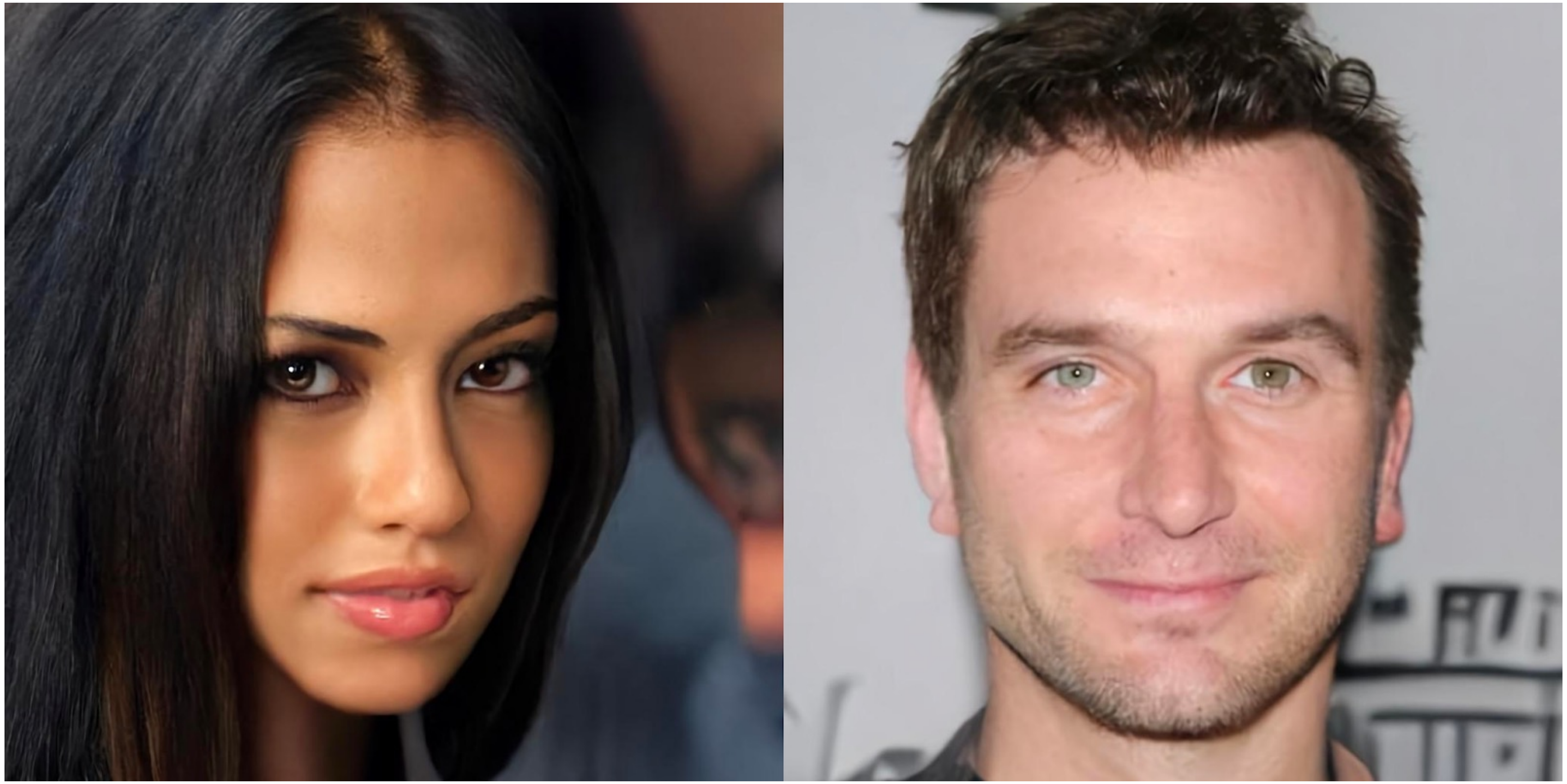


Think ‘infinite-layer’ latent variable model

[Tutorial by Calvin Luo and Yang Song]

Diffusion Models as Differential Equations

From discrete diffusion process to continuous diffusion process



[Tutorial by Calvin Luo and Yang Song]

Conditioning Diffusion Models

1. Directly training diffusion models with conditional information

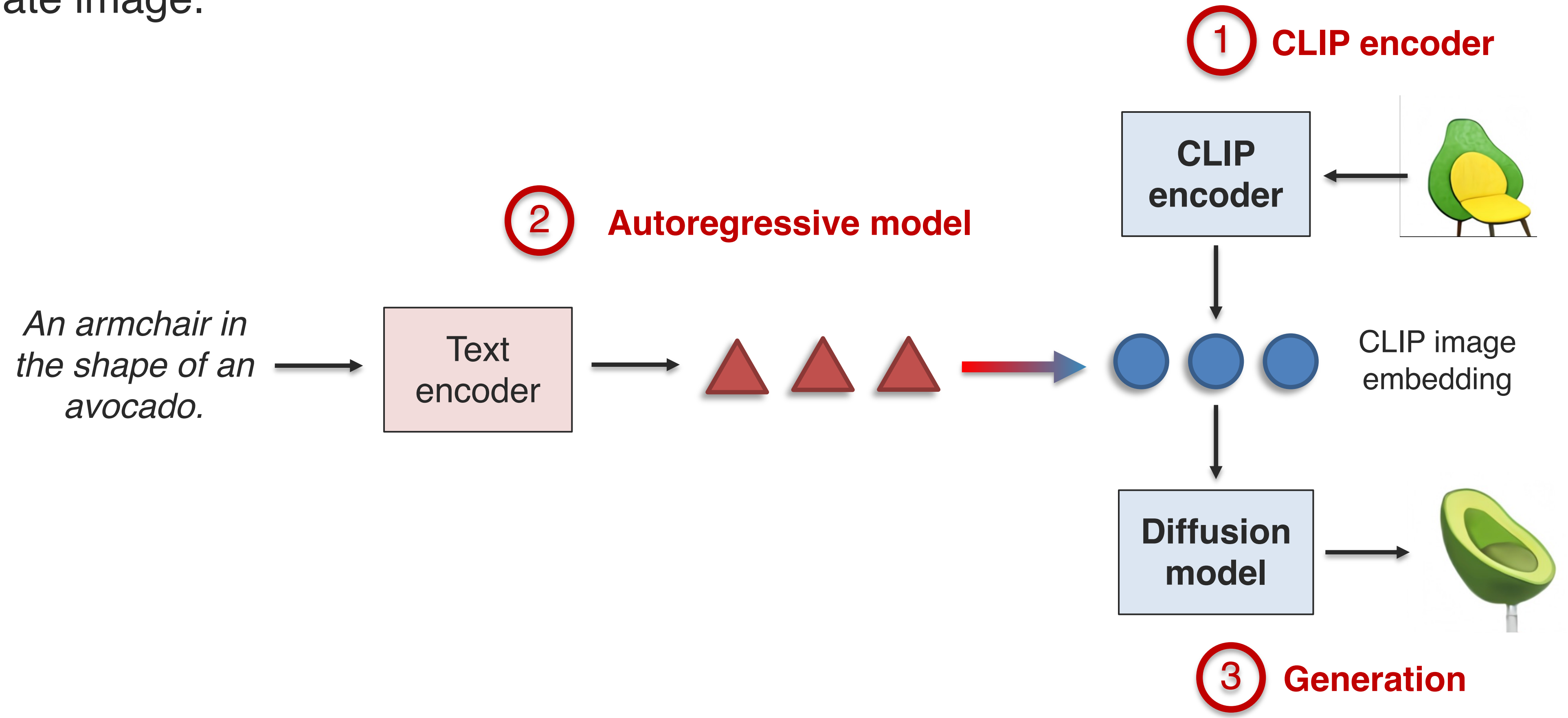
$$p(\boldsymbol{x}_{0:T}) = p(\boldsymbol{x}_T)\prod_{t=1}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t) \longrightarrow p(\boldsymbol{x}_{0:T} \mid y) = p(\boldsymbol{x}_T)\prod_{t=1}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, y)$$

1. Conditional original image prediction $\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t, y) \approx \boldsymbol{x}_0$
2. Conditional noise prediction $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t, y) \approx \boldsymbol{\epsilon}_0$
3. Conditional score function estimation $\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t, y) \approx \nabla \log p(\boldsymbol{x}_t \mid y)$

[Tutorial by Calvin Luo and Yang Song]

Text-to-Image Generation

1. Directly training diffusion models with conditional information
Conditional latent variables are pretrained CLIP embeddings, then diffusion model to generate image.



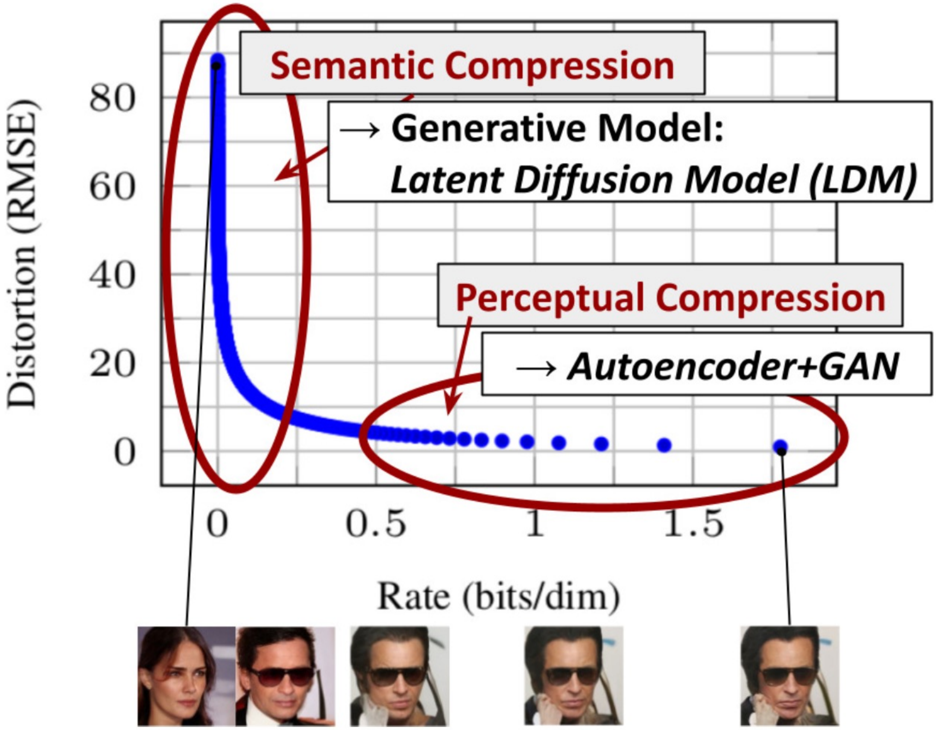
[Ramesh et al., Hierarchical Text-Conditional Image Generation with CLIP Latents. arXiv 2022]

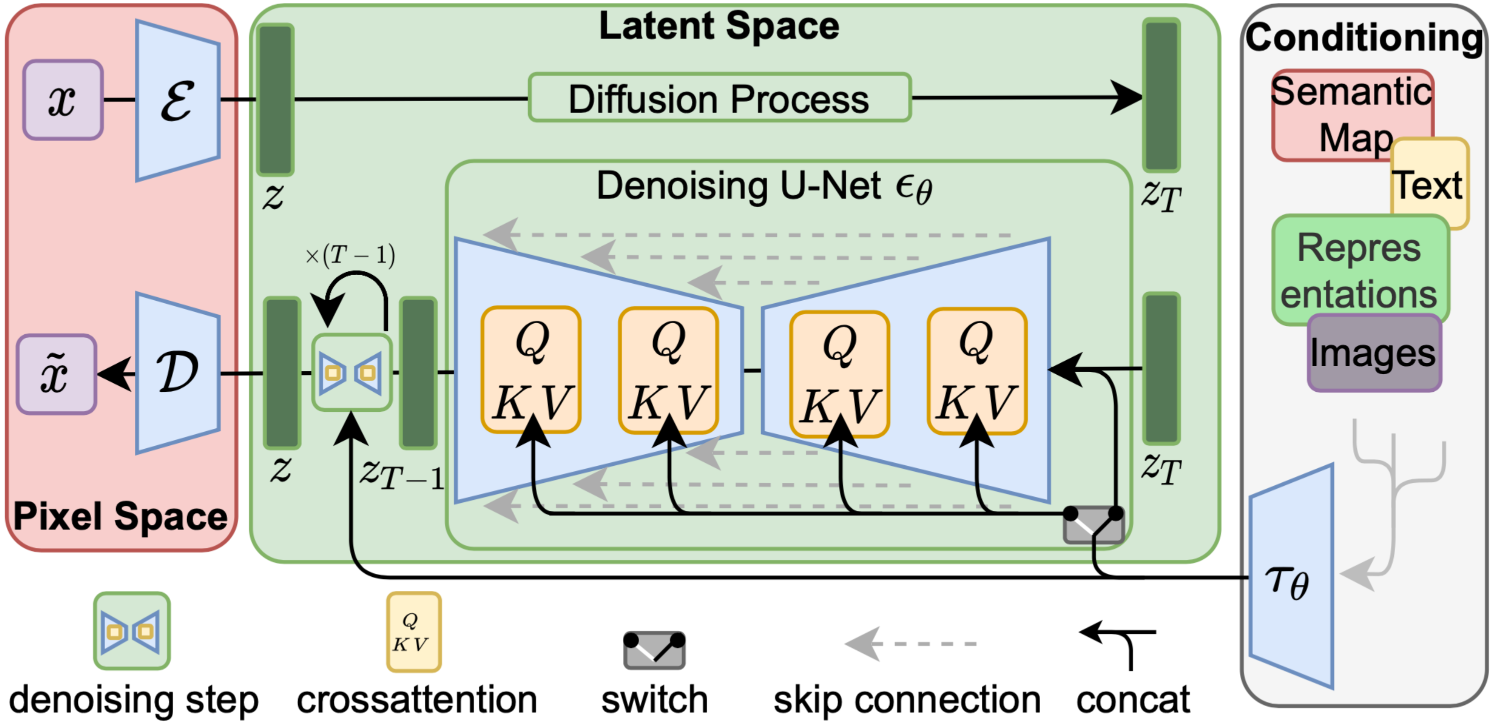
Text-to-Image Generation with Latent Diffusion

1. Directly training diffusion models with conditional information

Diffusion process in latent space instead of pixel space – faster training and inference.

Use autoencoder for perceptual compression, diffusion model for semantic compression.





[Rombach et al., High-Resolution Image Synthesis with Latent Diffusion Models. CVPR 2022]

Text-to-Image Generation with Latent Diffusion

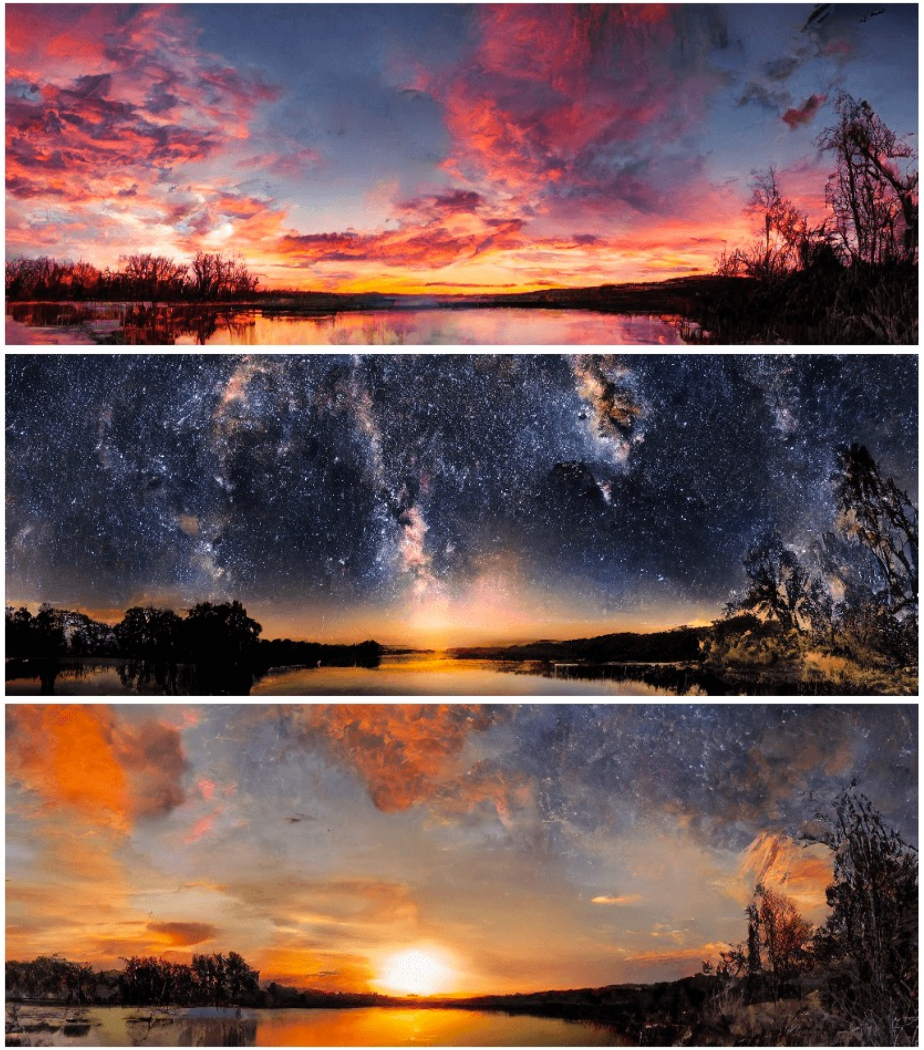
Text-to-Image Synthesis on LAION. 1.45B Model.

'A street sign that reads "Latent Diffusion" '	*'A zombie in the style of Picasso'*	*'An image of an animal half mouse half octopus'*	*'An illustration of a slightly conscious neural network'*	*'A painting of a squirrel eating a burger'*	*'A watercolor painting of a chair that looks like an octopus'*	*'A shirt with the inscription: "I love generative models!" '*

[Rombach et al., High-Resolution Image Synthesis with Latent Diffusion Models. CVPR 2022]

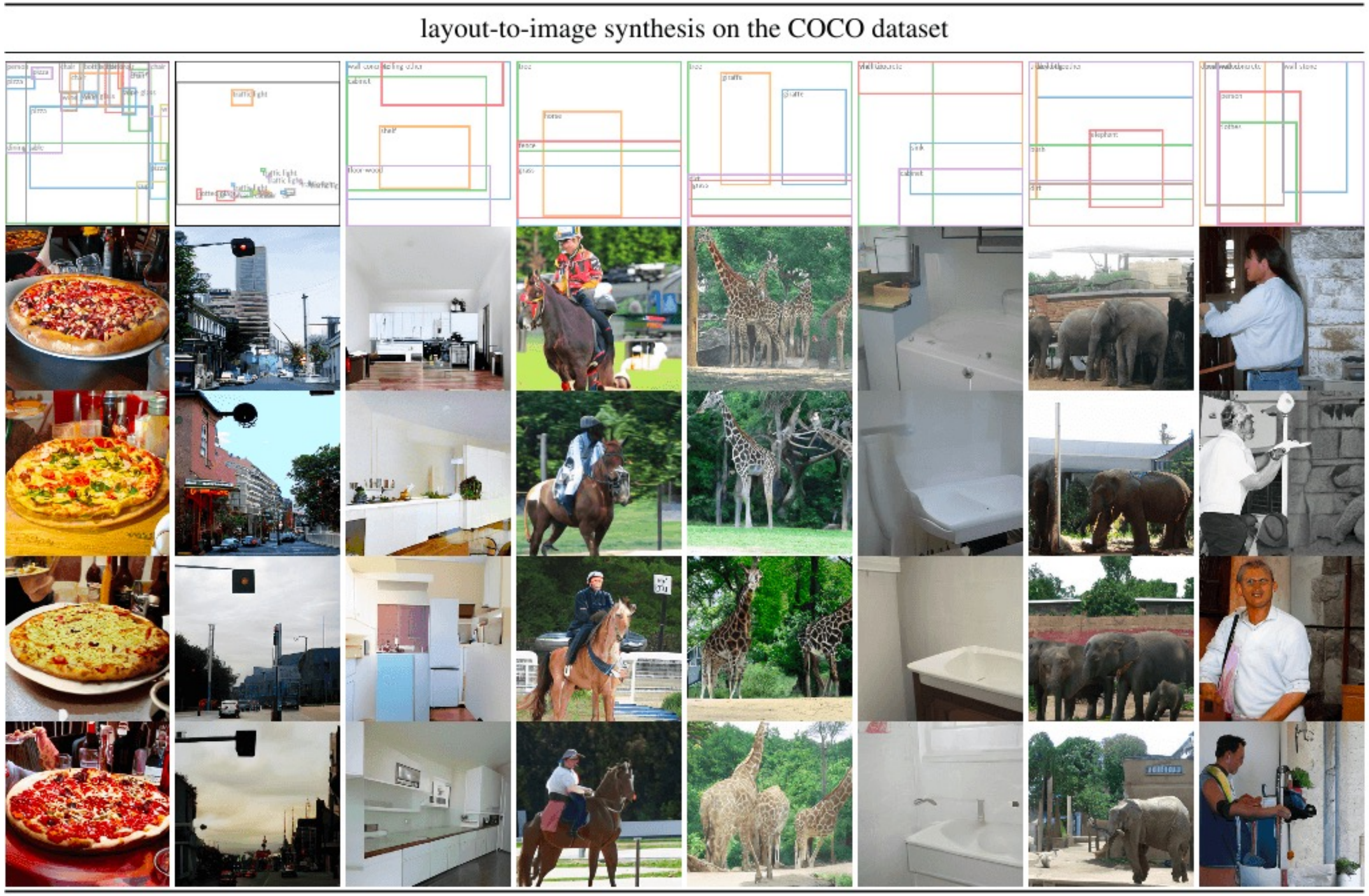
Text-to-Image Generation with Latent Diffusion

Semantic Synthesis on Flickr-Landscapes [21]



[Rombach et al., High-Resolution Image Synthesis with Latent Diffusion Models. CVPR 2022]

Text-to-Image Generation with Latent Diffusion



[Rombach et al., High-Resolution Image Synthesis with Latent Diffusion Models. CVPR 2022]

Conditioning Diffusion Models

2. Training unconditional diffusion model then classifier guidance

$$\nabla \log p(\boldsymbol{x}_t \mid y) = \underbrace{\nabla \log p(\boldsymbol{x}_t)}_{\text{unconditional score}} + \gamma \underbrace{\nabla \log p(y \mid \boldsymbol{x}_t)}_{\text{classifier gradient}}$$

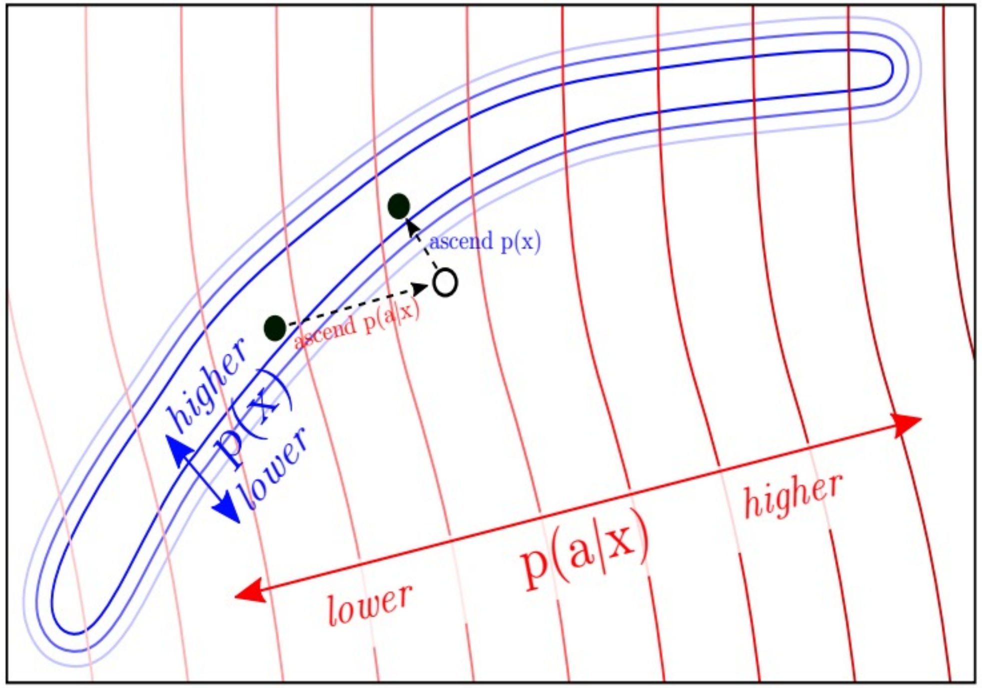
shiny → **y = +1**

$\boxed{\textbf{\textit{delicious}} \rightarrow \textbf{y = +1}}$

terrible → **y = -1**

Pretrained

The food tastes



[Tutorial by Calvin Luo and Yang Song]

Conditioning Diffusion Models

3. Training unconditional diffusion model then classifier-free guidance

$$
\begin{aligned}
\nabla \log p(\boldsymbol{x}_t \mid y) &= \nabla \log p(\boldsymbol{x}_t) + \gamma \left(\nabla \log p(\boldsymbol{x}_t \mid y) - \nabla \log p(\boldsymbol{x}_t)\right) \\
&= \nabla \log p(\boldsymbol{x}_t) + \gamma \nabla \log p(\boldsymbol{x}_t \mid y) - \gamma \nabla \log p(\boldsymbol{x}_t) \\
&= \underbrace{\gamma \nabla \log p(\boldsymbol{x}_t \mid y)}_{\text{conditional score}} + \underbrace{(1-\gamma) \nabla \log p(\boldsymbol{x}_t)}_{\text{unconditional score}}
\end{aligned}
$$

2 separate diffusion models, one conditional and one unconditional?

Just 1 diffusion model, unconditional training can be seen as setting y=constant

See empirical comparison by GLIDE paper – classifier-free guidance is more preferred

[Nichol et al., GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models. arXiv 2022]

Summary: Generative Models

Likelihood-based 1. VAEs – approximate inference via evidence lower bound	Fast & easy to train	Lower generation quality
2. Autoregressive models – exact inference via chain rule	Easy to train, exact likelihood	Slow to sample from
3. Flows – exact inference via invertible transformations	Easy to train, exact likelihood	Constrained architecture
4. Diffusion model – approximate inference via modeling noise	High generation quality	Slow to sample from
Likelihood-free 1. GANs – discriminative real vs generated samples	High generation quality	Hard to train, can't get features

Summary: Conditioning and Controlling Generative Models

1. Disentanglement

$$\mathcal{L}_\beta(x) = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - \beta \cdot \mathrm{KL}(q_\phi(z|x)||p(z))$$

2. Conditioning

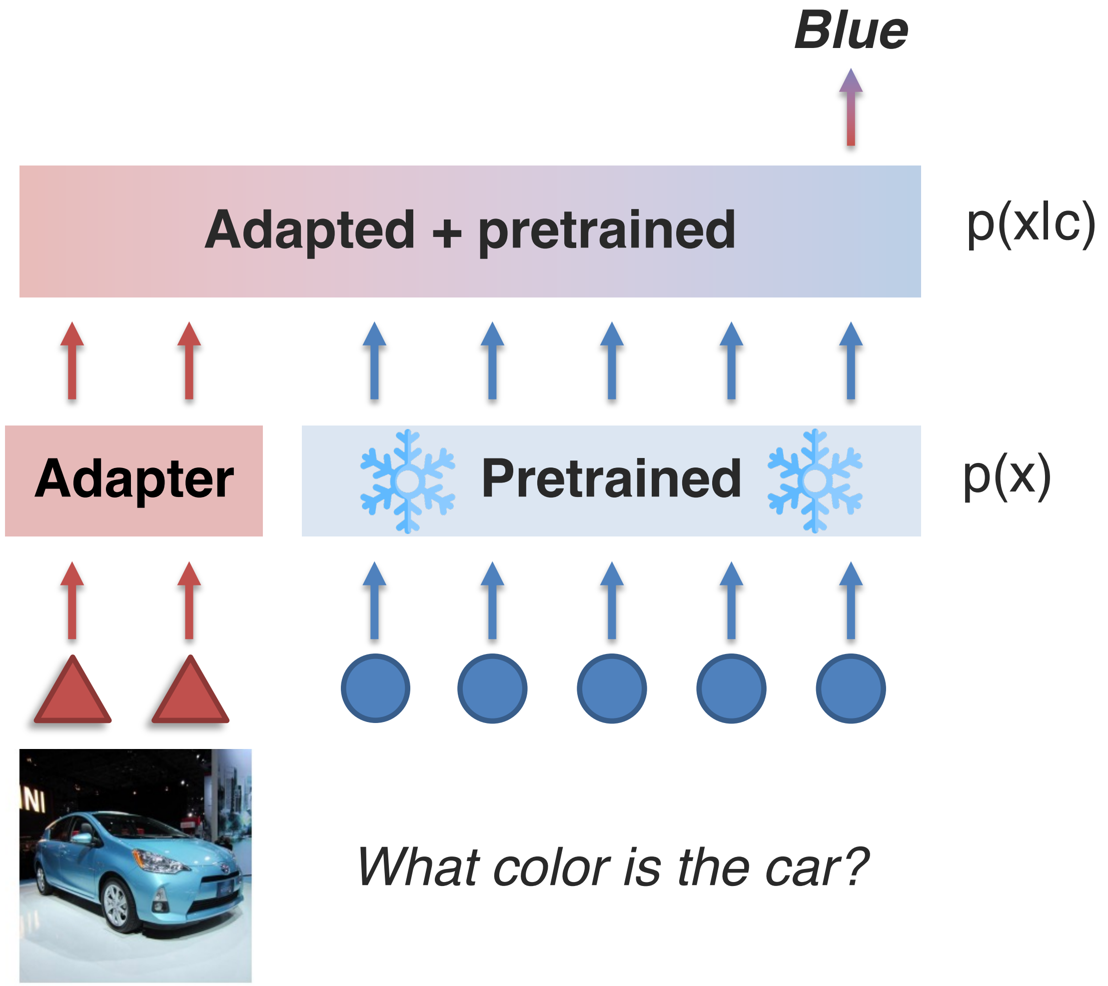
$$p(\boldsymbol{x}_{0:T} \mid y) = p(\boldsymbol{x}_T) \prod_{t=1}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, y)$$

Summary: Conditioning and Controlling Generative Models

1. Disentanglement

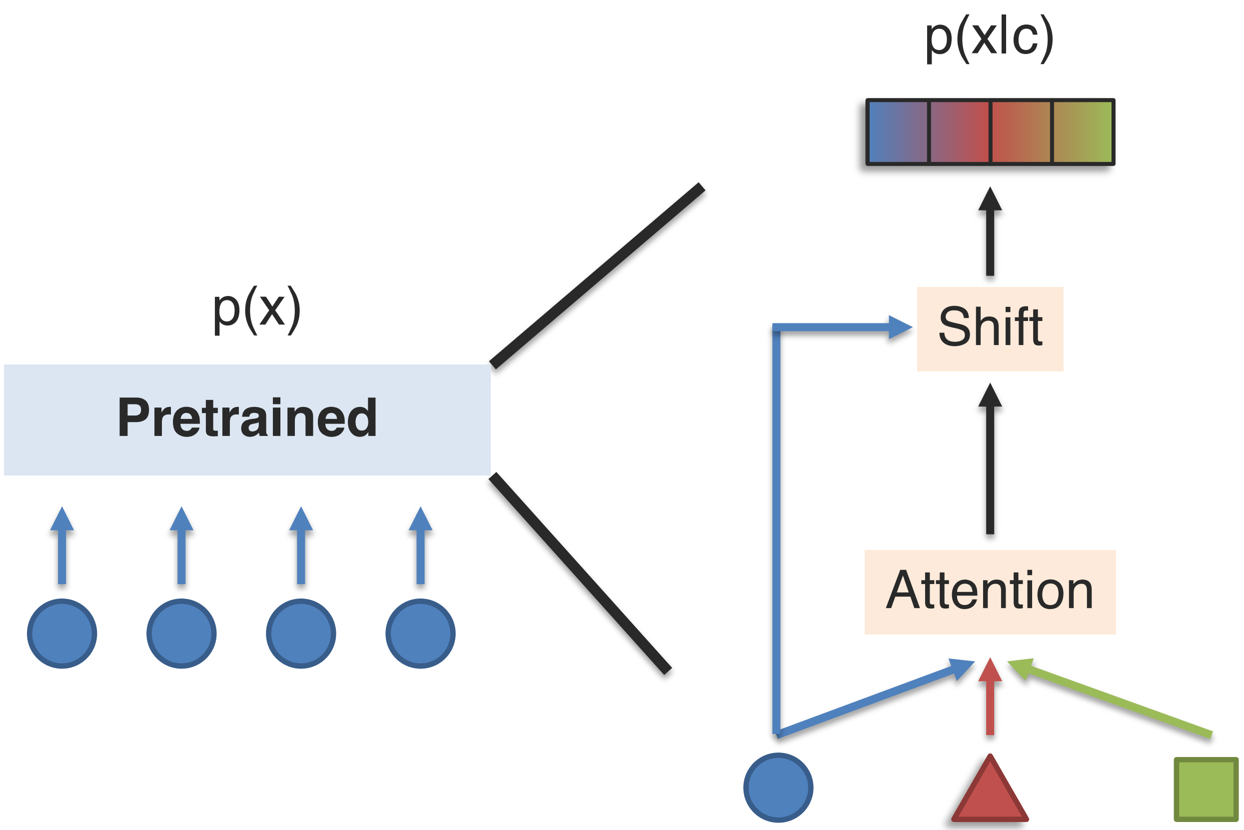
2. Conditioning

3. Prompt tuning



Summary: Conditioning and Controlling Generative Models

1. Disentanglement
2. Conditioning
3. Prompt tuning
4. Representation tuning



Summary: Conditioning and Controlling Generative Models

1. Disentanglement

$$\mathcal{L}_\beta(x) = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - \beta \cdot \mathrm{KL}(q_\phi(z|x)||p(z))$$

2. Conditioning

$$p(\boldsymbol{x}_{0:T} \mid y) = p(\boldsymbol{x}_T) \prod_{t=1}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, y)$$

3. Prompt tuning

4. Representation tuning

5. Classifier gradient tuning

$$\nabla \log p(\boldsymbol{x}_t \mid y) = \underbrace{\nabla \log p(\boldsymbol{x}_t)}_{\text{unconditional score}} + \gamma \underbrace{\nabla \log p(y \mid \boldsymbol{x}_t)}_{\text{classifier gradient}}$$

6. Classifier-free tuning

$$\nabla \log p(\boldsymbol{x}_t \mid y) = \underbrace{\gamma \nabla \log p(\boldsymbol{x}_t \mid y)}_{\text{conditional score}} + \underbrace{(1-\gamma) \nabla \log p(\boldsymbol{x}_t)}_{\text{unconditional score}}$$

Open Challenges

Open challenges

Definition: Simultaneously generating multiple modalities to increase information content while maintaining coherence within and across modalities.

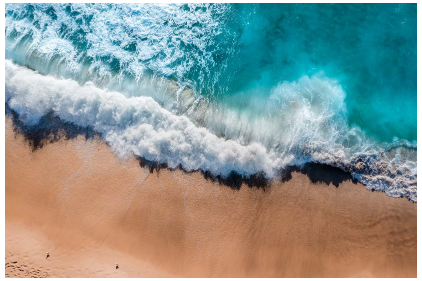
z_y

z_{a1}

z_{a2}

z_y

Many goals!



Cross-modal interactions

Recall representation & alignment!

Big dog on the beach. Waves crashing, people playing volleyball, …

Cross-modal interactions

'woof' *'crash'* *'bounce'* *'whoosh'*

Temporal + causal + logical structure

Recall reasoning!

Open Challenges

Open challenges

1. Synchronized generation over multiple modalities.
2. What's special about diffusion models from multimodal perspective?
2. Combining generation with explicit reasoning to enable compositional generation.
3. Better representation fusion and alignment in generation.
4. More control over large-scale generative models, fine-grained + few-shot control.
5. Human-centered evaluation of generative models.

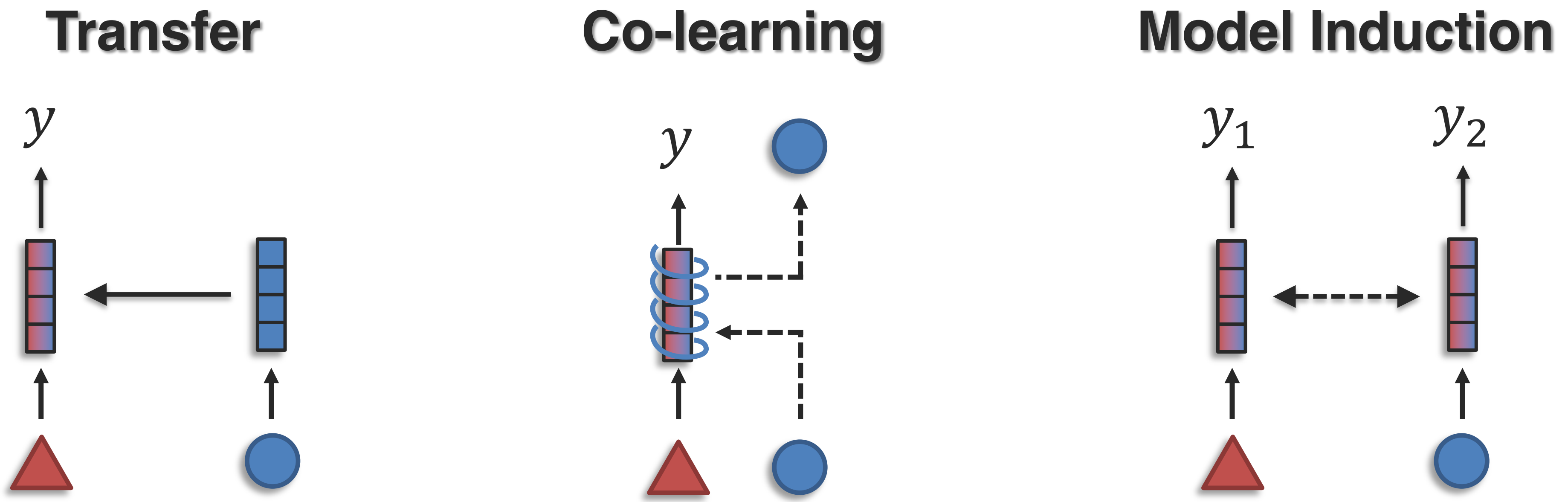
More resources:

https://lilianweng.github.io/tags/generative-model/
https://yang-song.net/blog/2021/score/
https://blog.evjang.com/2018/01/nf1.html & https://blog.evjang.com/2018/01/nf2.html
https://deepgenerativemodels.github.io/syllabus.html
https://www.cs.cmu.edu/~epxing/Class/10708-20/lectures.html
https://cvpr2022-tutorial-diffusion-models.github.io/
https://huggingface.co/blog/annotated-diffusion
https://calvinyluo.com/2022/08/26/diffusion-tutorial.html
https://jmtomczak.github.io/blog/1/1_introduction.html

Transference

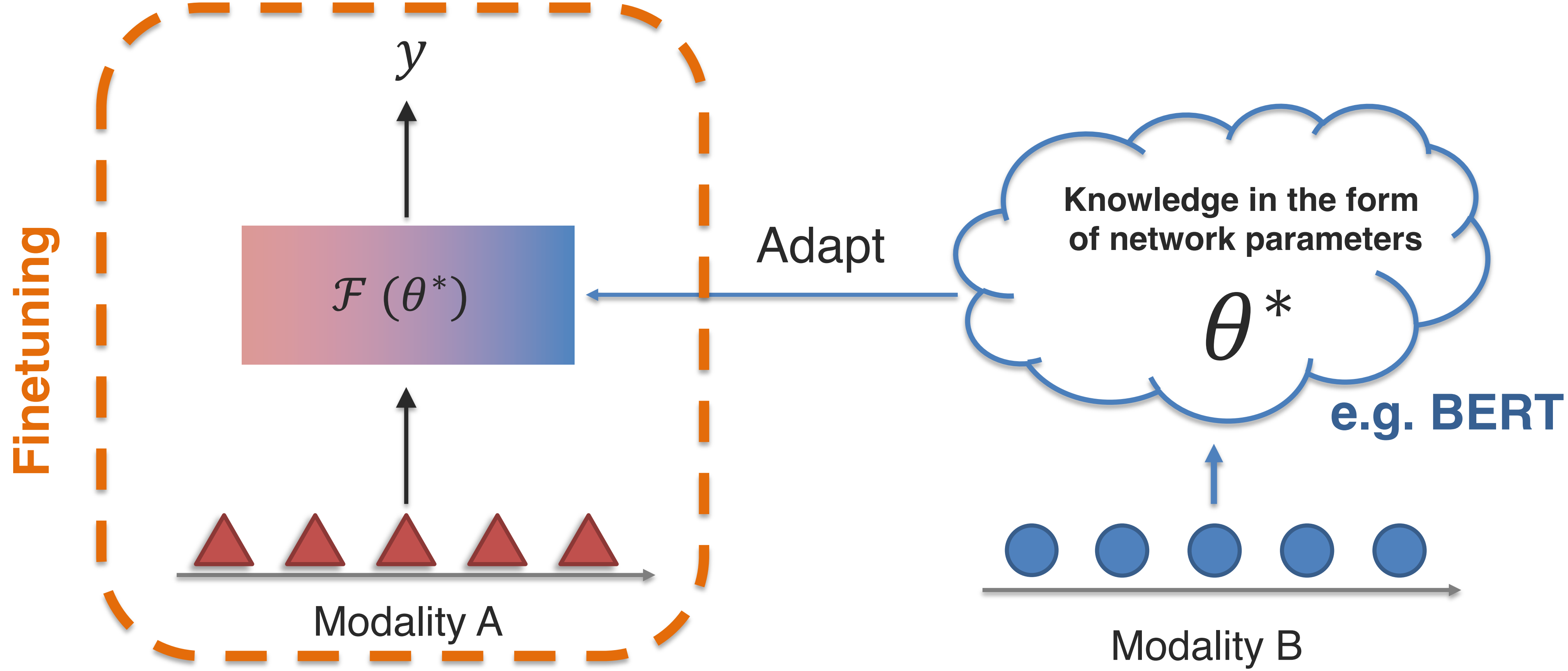
Definition: Transfer knowledge between modalities, usually to help the primary modality which may be noisy or with limited resources

Sub-challenges:



Sub-Challenge 5a: Transfer via Pretrained Models

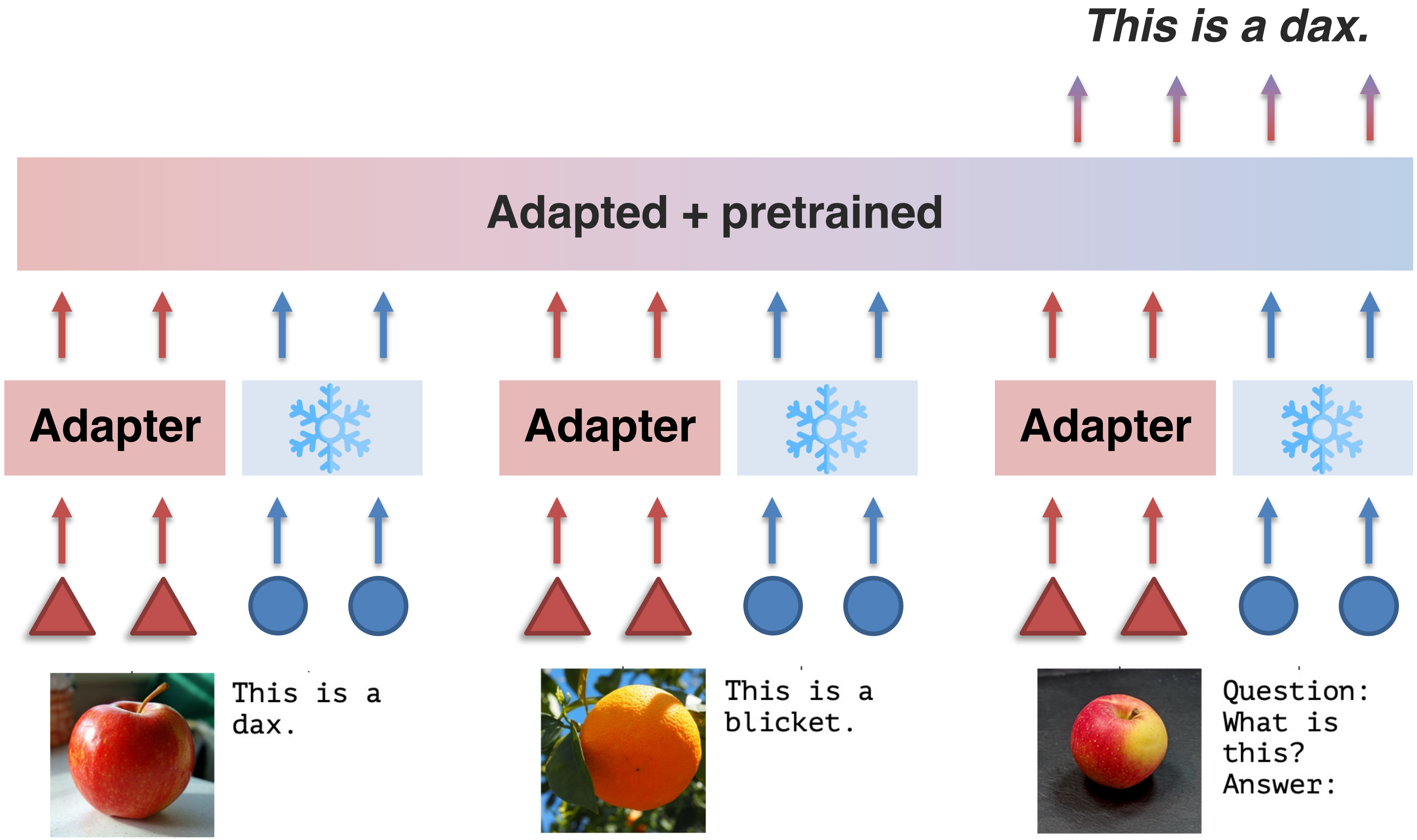
Definition: Transferring knowledge from large-scale pretrained models to downstream tasks involving the primary modality.



Sub-Challenge 5a: Transfer via Pretrained Models

Transfer via prefix tuning

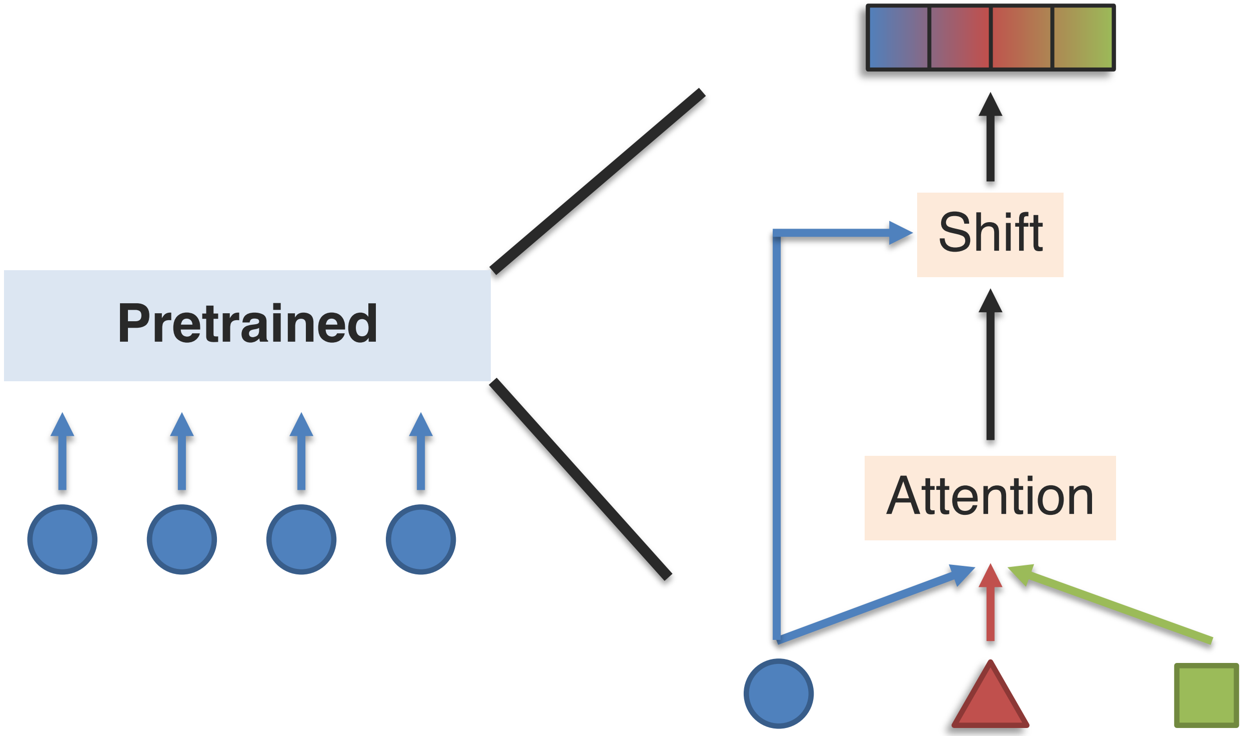
Few-shot image classification:

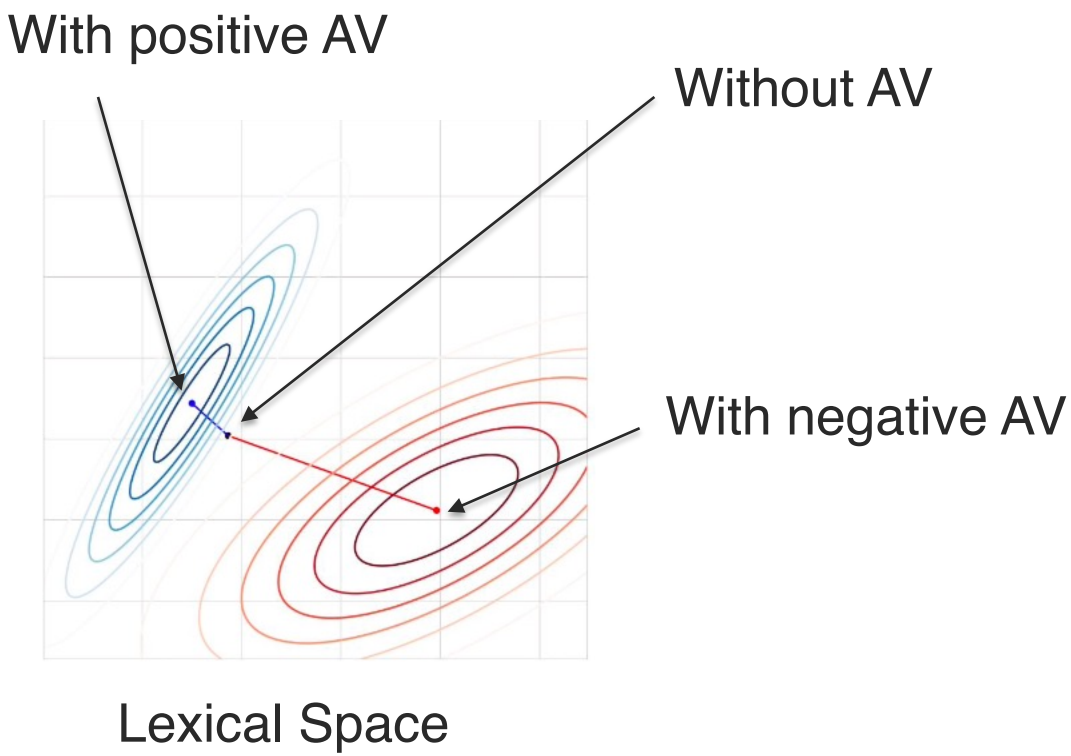


[Tsimpoukelli et al., Multimodal Few-Shot Learning with Frozen Language Models. NeurIPS 2021]

Sub-Challenge 5a: Transfer via Pretrained Models

Transfer via representation tuning





[Ziegler et al., Encoder-Agnostic Adaptation for Conditional Language Generation. arXiv 2019]

[Rahman et al., Integrating Multimodal Information in Large Pretrained Transformers. ACL 2020]

Sub-Challenge 5a: Transfer via Pretrained Models

1. Disentanglement

$$\mathcal{L}_\beta(x) = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - \beta \cdot \mathrm{KL}(q_\phi(z|x)||p(z))$$

2. Conditioning

$$p(\boldsymbol{x}_{0:T} \mid y) = p(\boldsymbol{x}_T) \prod_{t=1}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, y)$$

3. Prompt tuning

4. Representation tuning

5. Classifier gradient tuning

$$\nabla \log p(\boldsymbol{x}_t \mid y) = \underbrace{\nabla \log p(\boldsymbol{x}_t)}_{\text{unconditional score}} + \gamma \underbrace{\nabla \log p(y \mid \boldsymbol{x}_t)}_{\text{classifier gradient}}$$

6. Classifier-free tuning

$$\nabla \log p(\boldsymbol{x}_t \mid y) = \underbrace{\gamma \nabla \log p(\boldsymbol{x}_t \mid y)}_{\text{conditional score}} + \underbrace{(1-\gamma) \nabla \log p(\boldsymbol{x}_t)}_{\text{unconditional score}}$$