Lecture 9: Generative AI, Part 1

Yilun Du

CS 2281R: Mathematical & Engineering Principles for Training Foundation Models

Overview

- Introduction to Multimodal Generative Models
- Compositional Generative Models

Overview

- Introduction to Multimodal Generative Models
- Compositional Generative Models

Generative Models of Language

Can you construct the steps for language based intelligence?

Creating language-based intelligence involves several structured steps that combine data collection, model development, evaluation, and deployment. Here's a high-level framework:

1. Data Collection and Preparation

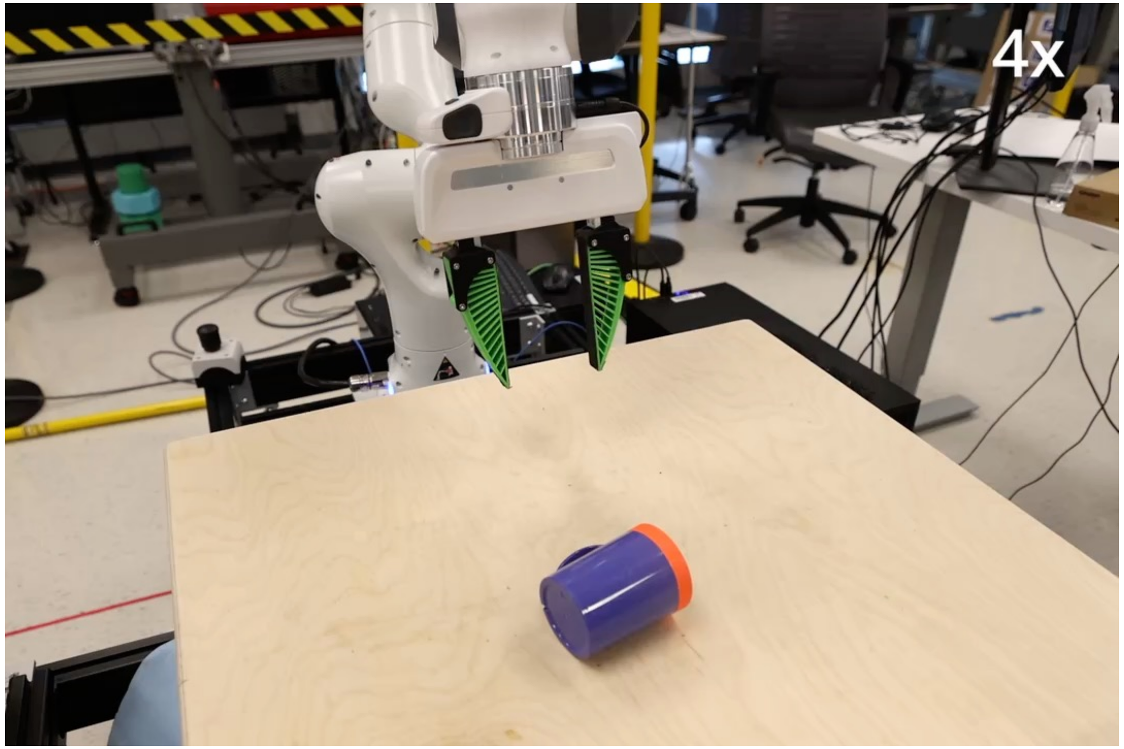
- **Data Acquisition**: Gather large datasets of text relevant to your use case, like books, articles, transcriptions, or domain-specific texts.
- **Data Cleaning**: Remove any irrelevant content, duplicates, special characters, and format inconsistencies.
- **Data Labeling**: If the task requires it (like sentiment analysis or classification), label the data with the appropriate tags or annotations.
- **Tokenization**: Break down text into manageable units (words, subwords, or characters) depending on the model's requirements.

Generative Models of Other Modalities

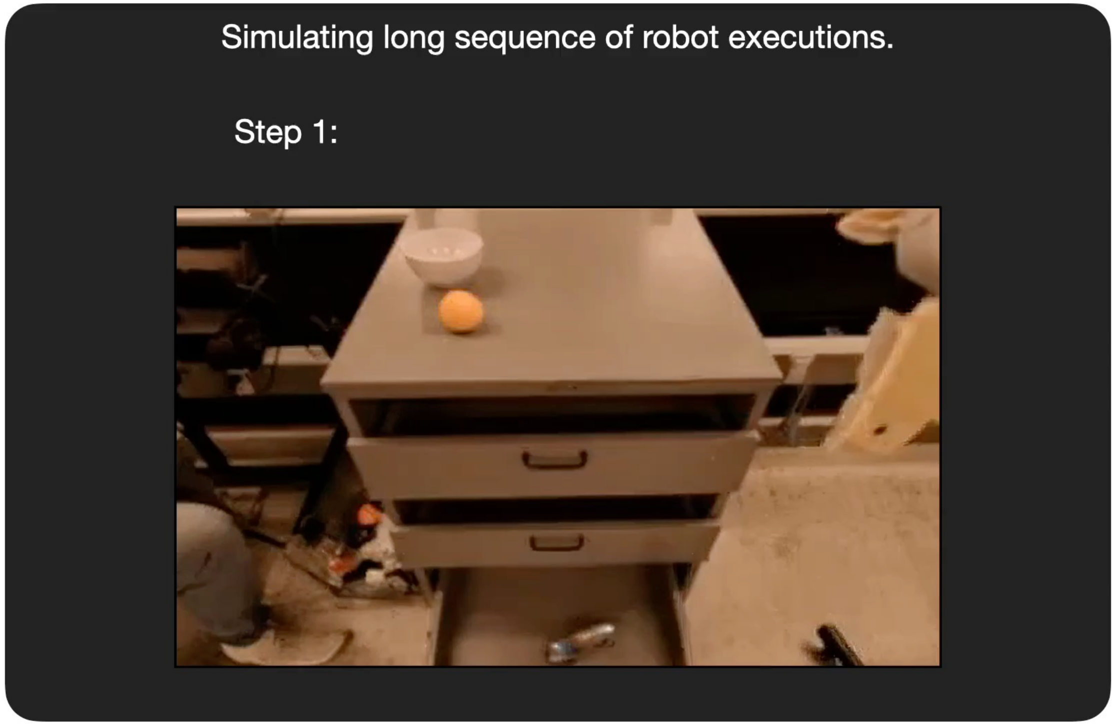
Images

Videos

Generative Models of Other Modalities



Actions



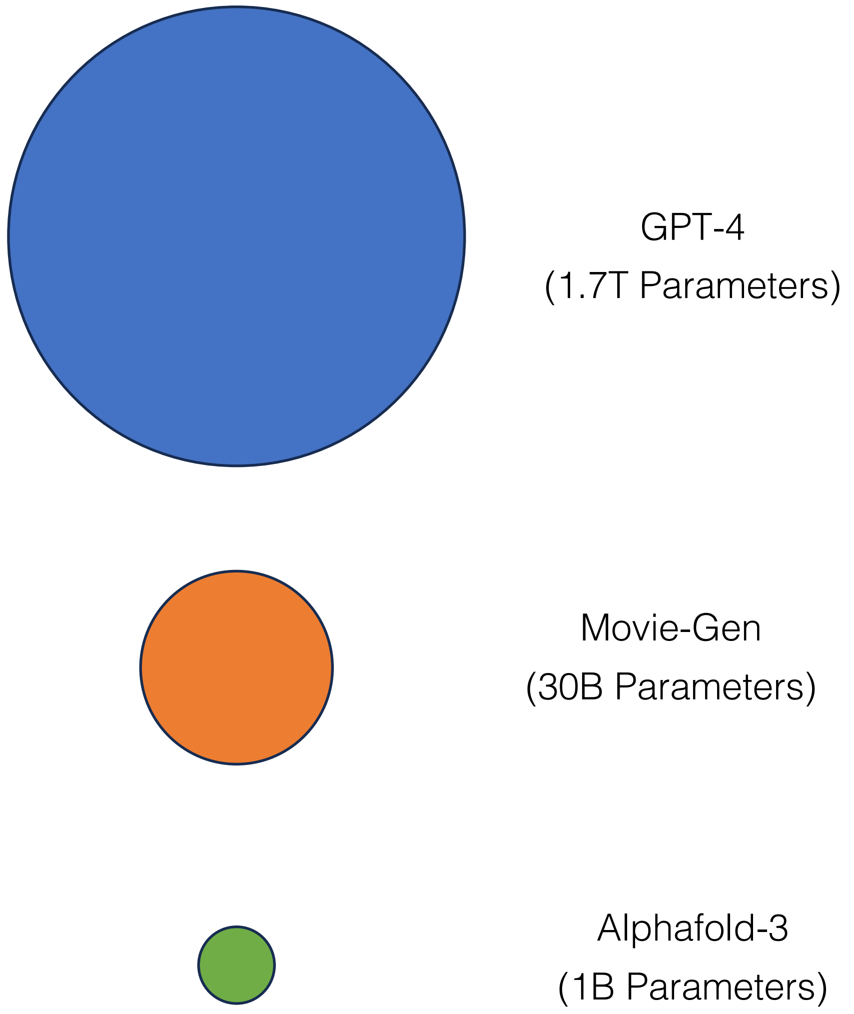
Simulation

What are Some Challenges of Other Modalities?

- Individual variables in the distribution are not necessarily autoregressively dependent on each other….
- Distributions are much higher dimensional than natural language, with much more uncertainty per pixel.
- We may not have data to cover the entire distribution we want to operate over.

Relative Sizes of Models

- Are at a much smaller size than that of language models (but are trained on more data!).
- Scaling laws are much weaker than those seen in language.
- Models are very frail – text-to-image models often fail to follow even simple text prompts that deviate from those seen in training.
- Why?
  - Language may be uniquely information-rich and compositional…



A Tale of Computational Scaling

DCGAN (2015)

BigGAN (2018)

Stable Diffusion (2022)

A Tale of Computational Scaling?

DCGAN (2015)

BigGAN (2018)

+ class conditioning
+ sample only high likelihood samples

Stable Diffusion (2022)

+ text conditioning
+ sample from distribution $p(x)(p(x|c)/p(x))^{\alpha}$
+ very careful data filtering

A Tale of Computational Scaling?

DCGAN (2015)

Model simple conditional distributions

BigGAN (2018)

$\boxed{\text{+ class conditioning}}$

+ sample only high likelihood samples

Stable Diffusion (2022)

$\boxed{\text{+ text conditioning}}$

+ sample from distribution $p(x)(p(x|c)/p(x))^{\alpha}$

+ very careful data filtering

A Tale of Computational Scaling?

DCGAN (2015)

BigGAN (2018)

Stable Diffusion (2022)

Generate samples from a modified probability distribution

+ class conditioning

+ sample only high likelihood samples

+ text conditioning

+ sample from distribution $p(x)(p(x|c)/p(x))^{\alpha}$

+ very careful data filtering

A Tale of Computational Scaling?

An astronaut riding a horse

Stable Diffusion (2022)

+ text conditioning
+ sample from distribution
$p(x)(p(x|c)/p(x))^{\alpha}$
+ very careful data filtering

A Tale of Computational Scaling?

An astronaut riding a horse

Stable Diffusion (2022)

+ text conditioning
+ sample from distribution
$p(x)(p(x|c)/p(x))^{\alpha}$
+ very careful data filtering

An astronaut riding a horse

Stable Diffusion (2022)

+ text conditioning
+ sample from distribution
$p(x)(p(x|c)/p(x))^{\alpha}$
+ very careful data filtering

A Tale of Computational Scaling?

An astronaut riding a horse

Stable Diffusion (2022)

+ text conditioning
+ sample from distribution $p(x)(p(x|c)/p(x))^\alpha$
+ very careful data filtering

An astronaut riding a horse

Stable Diffusion (2022)

+ text conditioning
+ sample from distribution $p(x)(p(x|c)/p(x))^\alpha$
+ very careful data filtering

A image

Stable Diffusion (2022)

+ text conditioning
+ sample from distribution $p(x)(p(x|c)/p(x))^\alpha$
+ very careful data filtering

A Tale of Computational Scaling?

An astronaut riding a horse

Stable Diffusion (2022)

An astronaut riding a horse

Stable Diffusion (2022)

A image

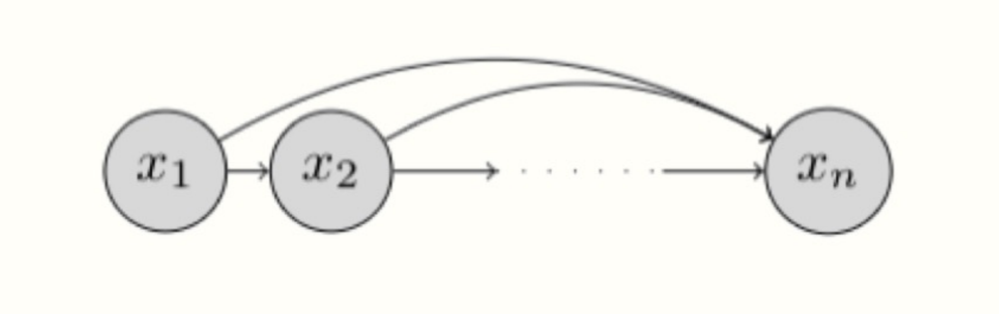
Stable Diffusion (2022)

Generative models cannot fit arbitrarily high dimensional distributions but rather ones that are simple (rich conditioning or low intrinsic dimensionality of data).

Autoregressive Generative Models

- Language models are typically parameterized as autoregressive generative models
  - This reflects the natural casual order of language
- Can construct generative models over other distributions in an autoregressive manner
  - But this does not necessarily follow the “structure of the domain”
  - In practice, we typically vector quantize continuous inputs into discrete tokens

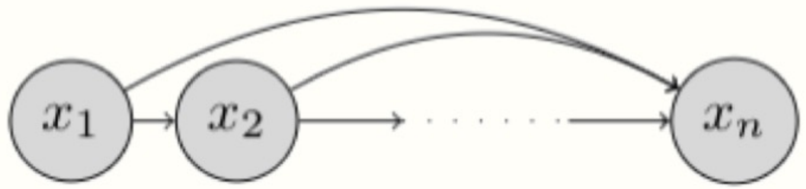
$$p(\mathbf{x}) = \prod_{i=1}^{n} p(x_i | x_1, x_2, \ldots, x_{i-1}) = \prod_{i=1}^{n} p(x_i | \mathbf{x}_{<i})$$



Limitations of Autoregressive Generative Models

- Learning an autoregressive factorization can be much harder than learning the base probability density itself:
  - For instance, consider a generative model of paths in a maze.
- The order of generation should follow the causal structure in the set of variables.
  - For example, any-order language models perform poorly
  - Generate pixels of an image in a hierarchical manner

$$p(\mathbf{x}) = \prod_{i=1}^{n} p(x_i | x_1, x_2, \ldots, x_{i-1}) = \prod_{i=1}^{n} p(x_i | \mathbf{x}_{<i})$$



Other Possible Generative Models

- There is a zoo of other generative models such as:
  - Energy-based Models
  - Variational Autoencoders
  - GANs
  - Flow Models
  - Diffusion models
  - Many of these generative model classes can be interconverted between each other

Energy Based Models

- The oldest class of generative models that inspired the development of many of the generative models we know today
  - Noise Contrastive Estimation -> GANs, Variational Partition Function Minimization -> VAEs, Ancestral Importance Sample + Score Matching -> Diffusion
- Represent the probability distribution $p_\theta(x)$ as an unnormalized distribution parameterized with an energy function $E_\theta(x)$ where $p_\theta(x) \propto e^{-E_\theta(x)}$
  - Allows us to represent any probability distribution with a neural network that outputs a single scalar output
- The simplest training objective for EBMs is approximate maximum likelihood estimation

Energy Based Models

- The likelihood of datapoint x is given under an EBM is give by $p_\theta(x) = \ e^{-E_\theta(x)} / \int e^{-E_\theta(x)} dx$, where denominator is known as the partition function and is usually intractable to compute.
- The gradient of maximum likelihood training for a point x is given by:

$$
\begin{aligned}
\nabla_\theta \log p_\theta(x) &= -\nabla_\theta E_\theta(\mathrm{x}) - \nabla_\theta \log \int e^{-E_\theta(x)} dx \\
&= -\nabla_\theta E_\theta(\mathrm{x}) + \frac{\int \nabla_\theta E_\theta(x) e^{-E_\theta(x)} dx}{\int e^{-E_\theta(x)} dx} \\
&= -\nabla_\theta E_\theta(\mathrm{x}) + \mathbb{E}_{x \sim p_\theta(x)}[\nabla_\theta E_\theta(x)]
\end{aligned}
$$

- The last expression can estimate through Monte Carlo approximation by sampling from the model distribution!

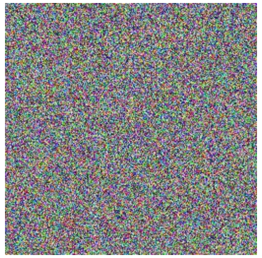
Energy Based Models

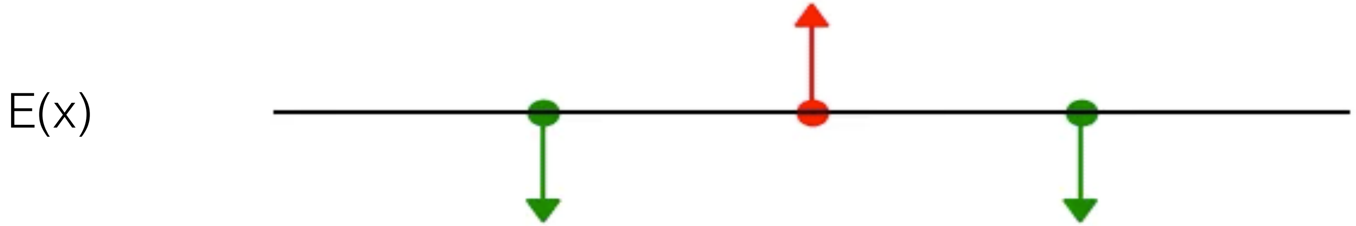
- The likelihood of datapoint x is given under an EBM is give by $p_\theta(x) = e^{-E_\theta(x)} / \int e^{-E_\theta(x)} dx$, where denominator is known as the partition function and is usually intractable to compute.
- This corresponds to maximum likelihood loss:

$$\nabla_\theta \mathcal{L}_{ML} = \underbrace{\mathbb{E}_{x\sim p(x)}}_{\text{Drawn from Data}}[\nabla_\theta E(x)] - \underbrace{\mathbb{E}_{x\sim p_\theta(x)}}_{\text{Drawn from Learned Distribution}}[\nabla_\theta E(x)]$$

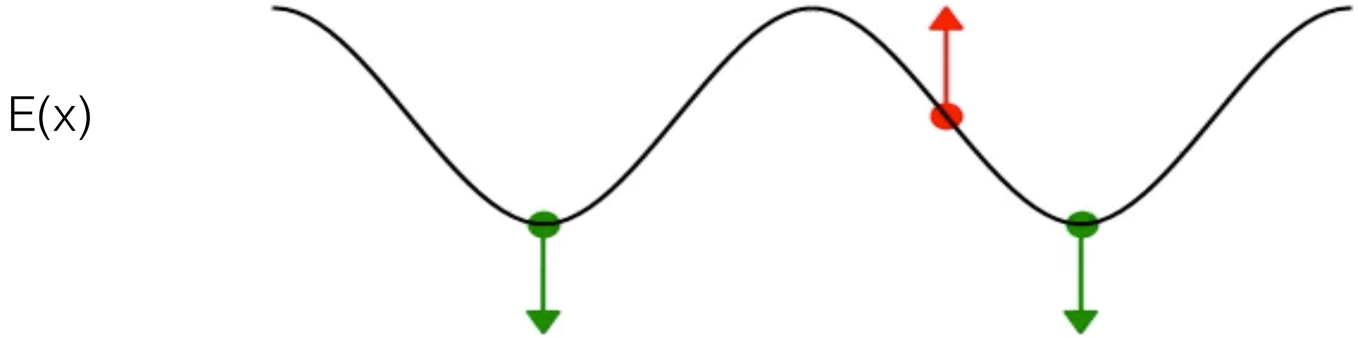
- Contrastively decrease the energy of real data while increase the energy of samples from the model (inspired the development of contrastive learning).

Learning Energy Functions

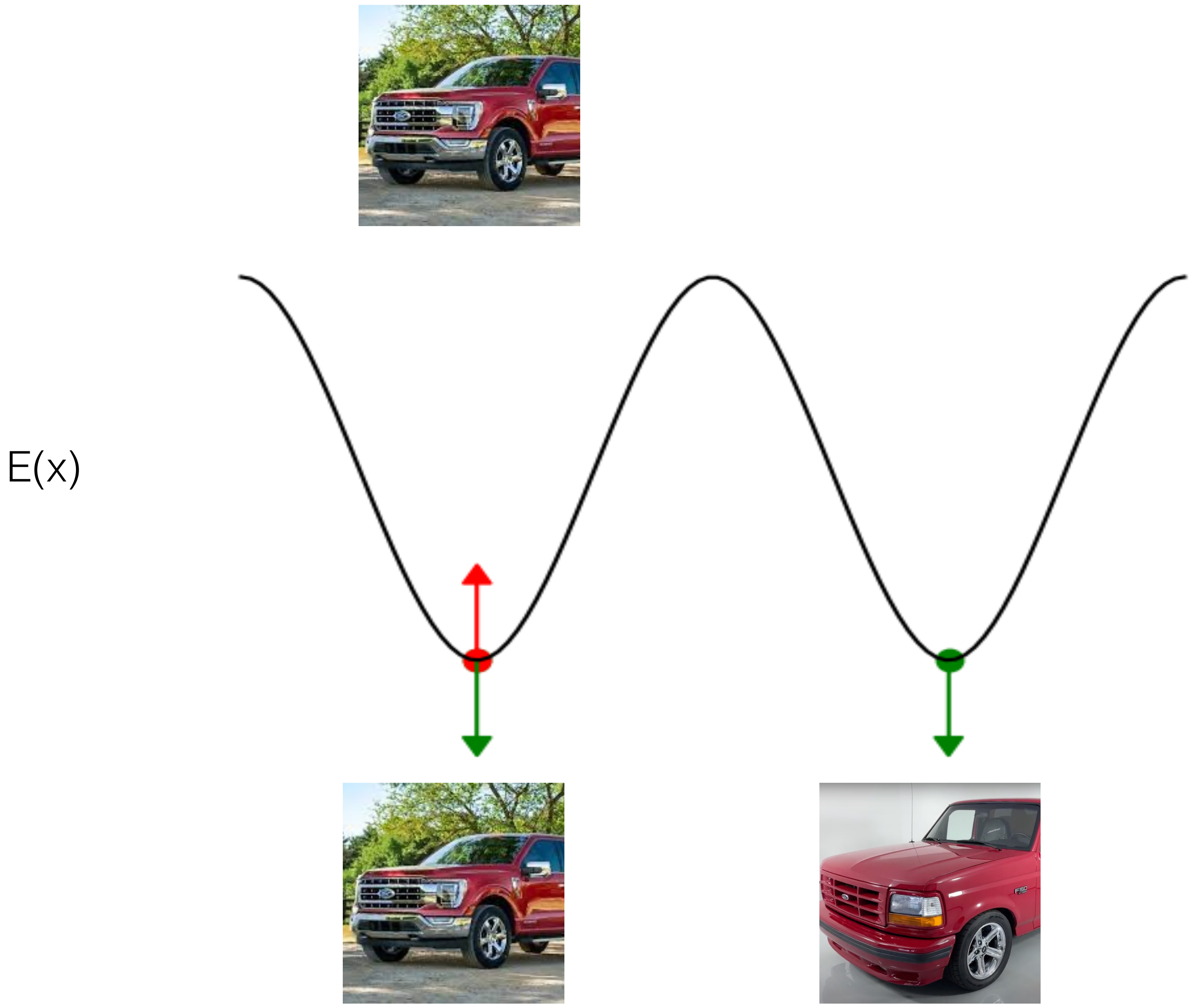




Learning Energy Functions



Learning Energy Functions



Limitations of Energy Based Models

- EBMs make training very challenging and slow because it requires you to explicitly draw MCMC sample from the model probability distribution in order to train the model
  - This inspired many future generative models which learned explicit networks for sampling
- However, this comes with benefits. – the parameterization $p_\theta(x) \propto e^{-E_\theta(x)}$ makes no assumptions on the nature of the probability distribution modeled, allowing the model to learn to flexibly capture any distribution
- In contrast, all other generative models make assumptions on the structure of the probability distribution they are modeling, which can be inaccurate dependent on the distribution.

From EBMs to Variational Autoencoders

- Training EBMs is challenging because it involves sampling from a high dimensional distribution $p_\theta(x)$ for maximum likelihood training
- We can make it easier by factorizing the distribution with a latent distribution

$$p_\theta(x) = \int p_\theta(x|z)p(z)dz$$

- Learning $p_\theta(x|z)$ can be much easier than learning $p_\theta(x)$ -- for instance $p_\theta(x|z)$ can be Gaussian even when $p_\theta(x)$ is not
- However, maximum likelihood training of $p_\theta(x)$ still requires us to exhaustively sample all value of $p(z)$ which is intractable

From EBMs to Variational Autoencoders

- Use variational inference to learn an amortized sampler $q_{\emptyset}(z|x)$ for $p(z)$ given an input x

$$p_\theta(x) = \int p_\theta(x|z)p(z)dz = \int p_\theta(x|z)p(z)\frac{q_{\emptyset}(z|x)}{q_{\emptyset}(z|x)}dz = \mathrm{E}_{q_{\emptyset}(x|z)}\left[p(z)\frac{p_\theta(x|z)}{q_{\emptyset}(z|x)}\right]$$

- Using the Jensen's inequality, we can write the log-likelihood as:

$$\log p_\theta(x) = \log\left(\mathrm{E}_{q_{\emptyset}(z|x)}\left[p(z)\frac{p_\theta(x|z)}{q_{\emptyset}(x|z)}\right]\right) \geq \mathrm{E}_{q_{\emptyset}(z|x)}[p_\theta(x|z)] - KL(q_{\emptyset}(z|x)||p(z))$$

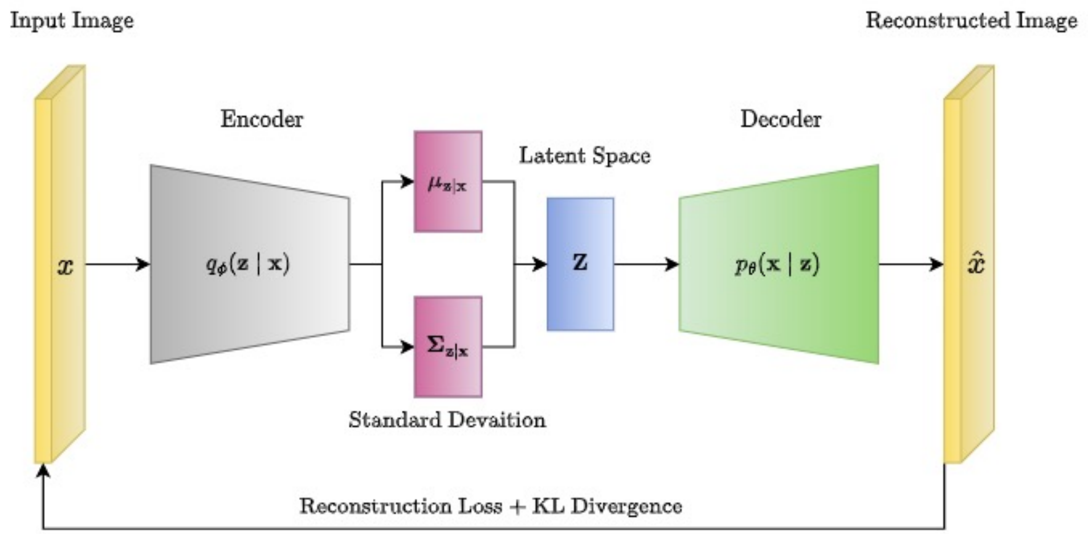
- The above objective is exactly the training objective used to train the VAE!

Variational Autoencoders

- Maximize log-likelihood in the form

$$\log p_\theta(x) = \log\left(\mathrm{E}_{q_\emptyset(z|x)}\left[p(z)\frac{p_\theta(x|z)}{q_\emptyset(x|z)}\right]\right) \geq \mathrm{E}_{q_\emptyset(z|x)}[p_\theta(x|z)] - KL(q_\emptyset(z|x)||p(z))$$

- In a VAE, we represent $p_\theta(x|z)$ and $q_\emptyset(z|x)$ as Gaussian distributions



From Variational Autoencoders / EBMs to Diffusion Models

- In practice, VAEs often generate blurry samples as both amortized sampler and generator have limited capacity
- Reduce the capacity requirements for each component of the variational procedure by constructing an annealed sequence of intermediate latent variables (inspired from anneal importance sampling from EBMs)

$$p\left(\mathbf{x}^{(0\cdots T)}\right) = p\left(\mathbf{x}^{(T)}\right) \prod_{t=1}^{T} p\left(\mathbf{x}^{(t-1)}|\mathbf{x}^{(t)}\right).$$

$$\log\ p\left(\mathbf{x}^{(0\cdots T)}\right) = \log\left[\begin{array}{c} \int d\mathbf{x}^{(1\cdots T)} q\left(\mathbf{x}^{(1\cdots T)}|\mathbf{x}^{(0)}\right) \cdot \\ p\left(\mathbf{x}^{(T)}\right) \prod_{t=1}^{T} \frac{p\left(\mathbf{x}^{(t-1)}|\mathbf{x}^{(t)}\right)}{q\left(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)}\right)} \end{array}\right]$$

Diffusion Models

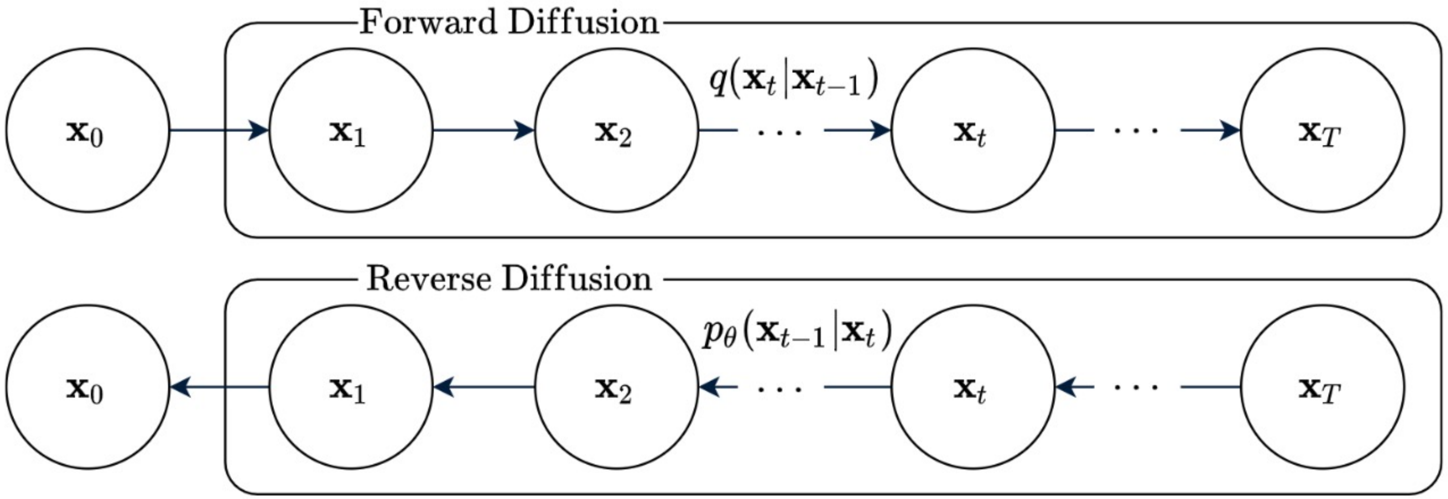
- In practice, VAEs often generate blurry samples as both amortized sampler and generator have limited capacity
- Reduce the capacity requirements for each component of the variational procedure by constructing an annealed sequence of intermediate latent variables (developed originally to draw samples from the partition function of EBMs)

$$\mathbb{E}\left[-\log p_\theta(\mathbf{x}_0)\right] \le \mathbb{E}_q\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] = \mathbb{E}_q\left[-\log p(\mathbf{x}_T) - \sum_{t\geq 1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] =: L$$

$$\mathbb{E}_q\left[\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0)\parallel p(\mathbf{x}_T))}_{L_T} + \sum_{t>1}\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)\parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}} \underbrace{- \log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0}\right]$$

Diffusion Models

- In practice, VAEs often generate blurry samples as both amortized sampler and generator have limited capacity
- Reduce the capacity requirements for each component of the variational procedure by constructing an annealed sequence of intermediate latent variables (developed originally to draw samples from the partition function of EBMs)

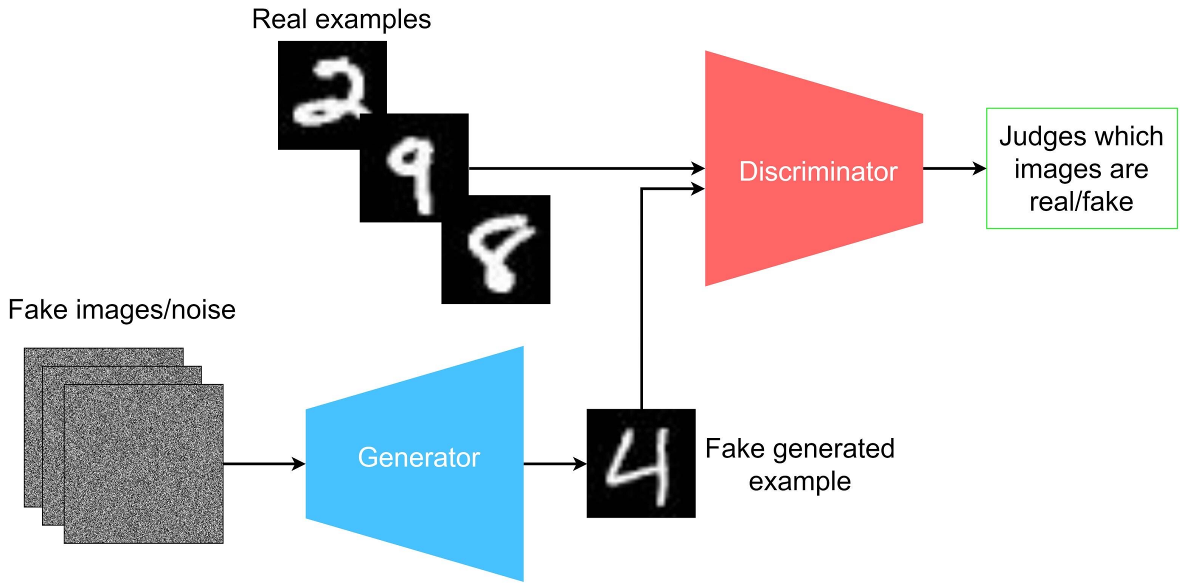


From EBMs to GANs

- To learn an EBM $p_\theta(x) \propto e^{-E_\theta(x)}$ to fit a probability distribution $p_D(x)$, one clever way is through classification (noise contrastive estimation)
- Given samples from a data distribution $p_D(x)$ and a noise distribution $p_{noise}(x)$, we can implicitly recover the energy function $E_\theta(x)$ by training a classifier $\mathcal{H}(x)$ classifying if a data point is either from the data or noise distribution. The energy function is the difference of the logits $\mathcal{H}(x)$ and $p_{noise}(x)$.
- This procedure allows us to replace the difficult problem of drawing samples from model distribution with generating samples from the noise distribution. However, the variance of this estimator still depends on how close $p_{noise}(x)$ is to sampling from the $p_\theta(x)$

GANs

- Instead of explicitly constructing a distribution $p_{noise}(x)$, learn the neural network g(z) that approximates this distribution!
- Continue using the classifier $\mathcal{H}(x)$ to distinguish between real and fake samples
- The noise distribution is the generator and the classifier is the discriminator, leading to GANs!



Overview

- Introduction to Multimodal Generative Models
- Compositional Generative Models

Compositional Generative Models

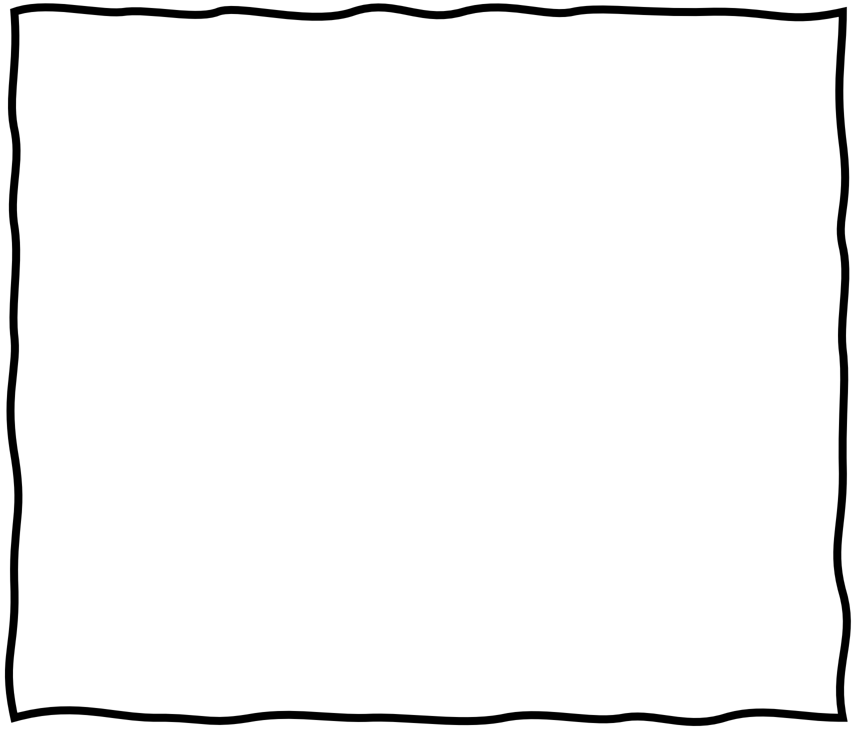
- In practice, in many multimodal domains, data is scarce and often covers a sparse subset of the distribution we would like generative models to operate over.
- We can use the idea of compositional generative modeling to induce models to generate samples that are outside of the distribution of the data seen by the model

$$p(x, y, z) = p(x, y)\, p(y, z)$$

- Above factorization only requires data to be seen from the pairwise marginals instead of the full joint distribution of samples

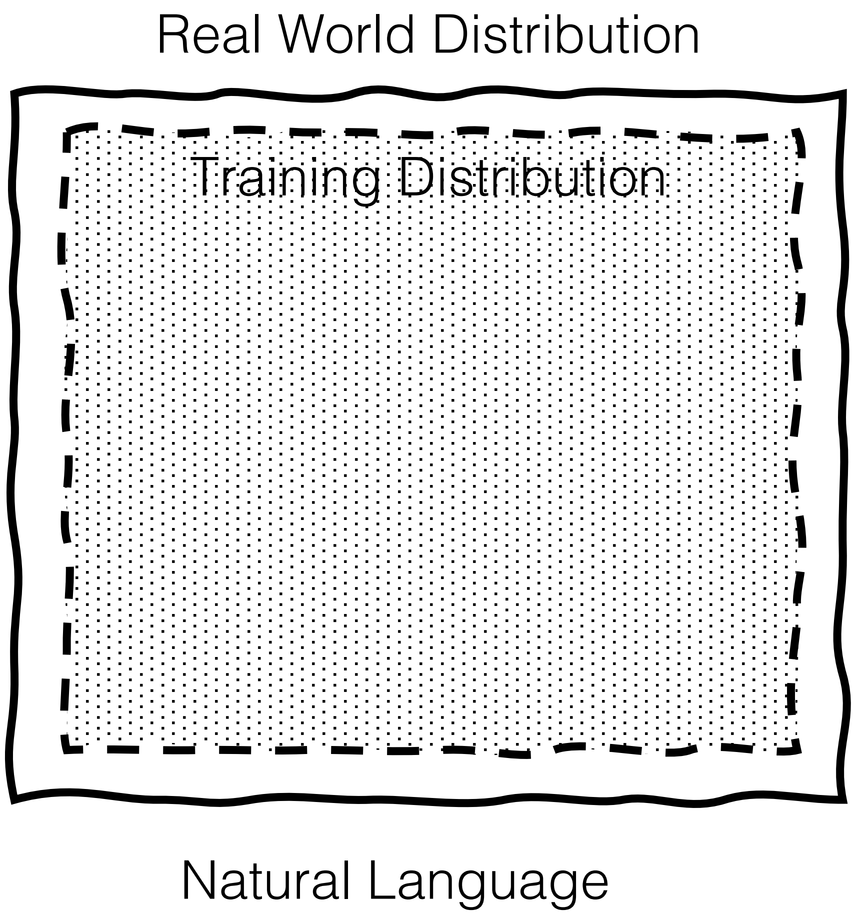
Why Does Generative AI for Language Work So Well?

Real World Distribution



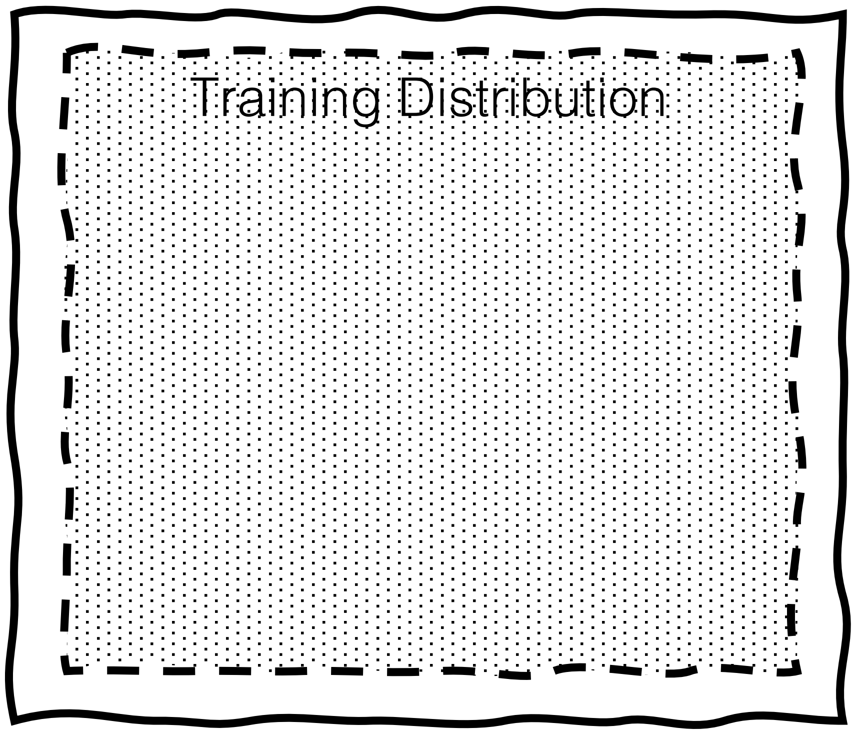
Natural Language

Why Does Generative AI for Language Work So Well?



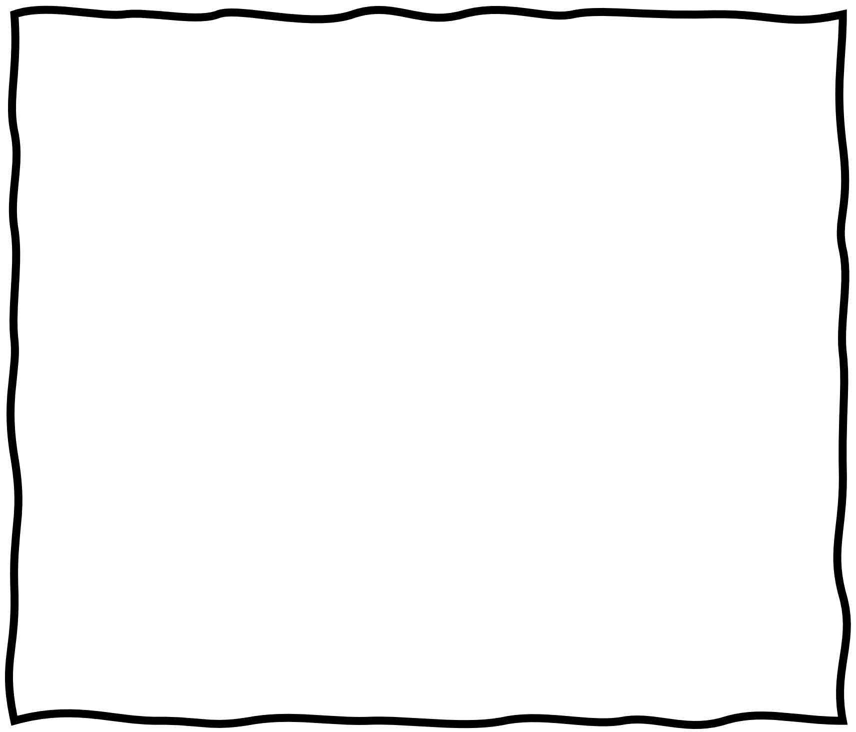
Why Does Generative AI on Other Settings Work Poorly?

Real World Distribution



Natural Language

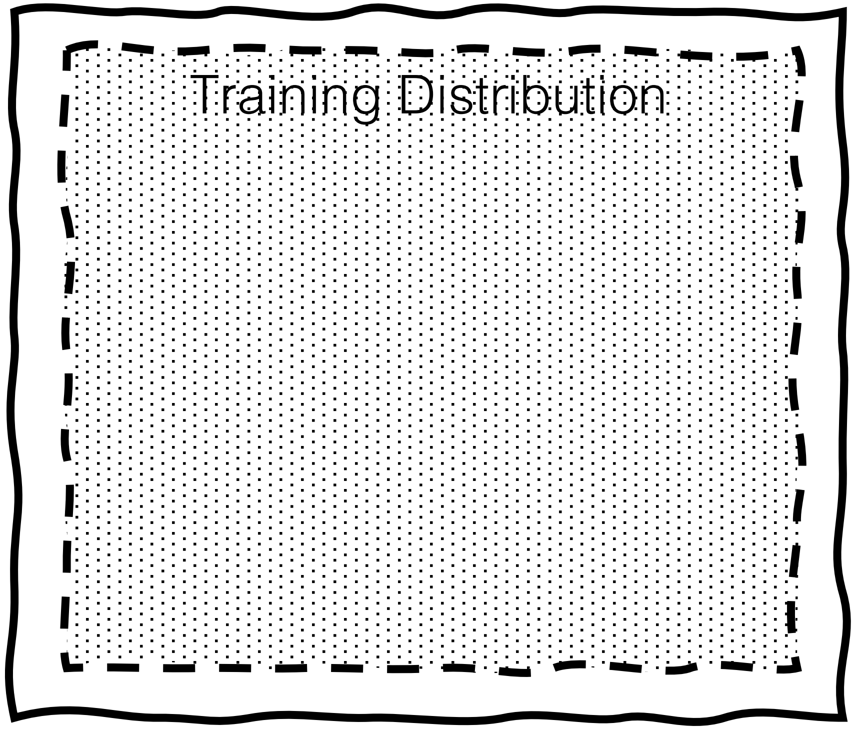
Real World Distribution



Embodied Data

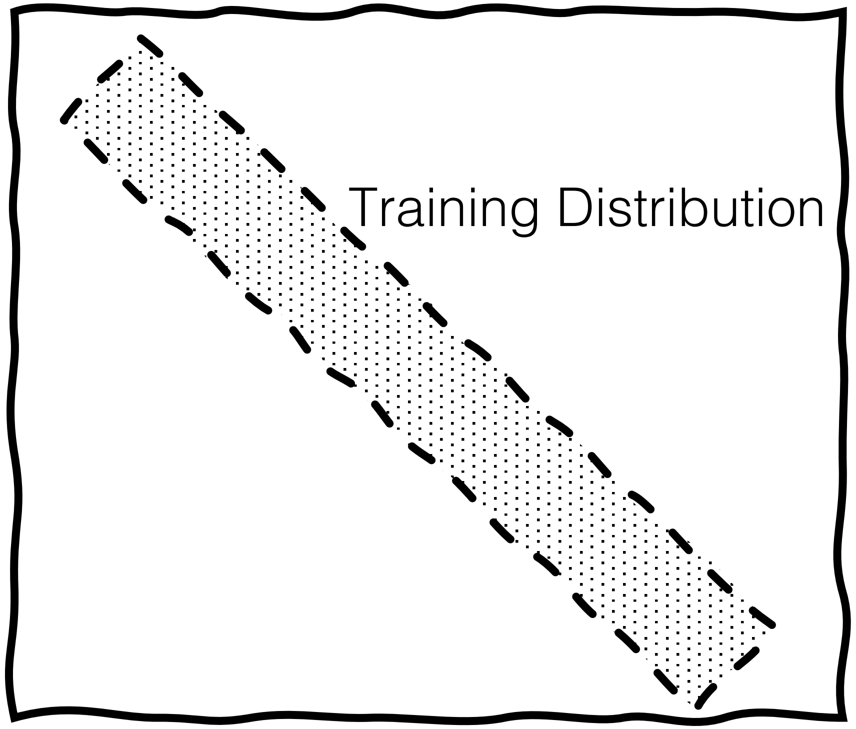
Why Does Generative AI on Other Settings Work Poorly?

Real World Distribution



Natural Language

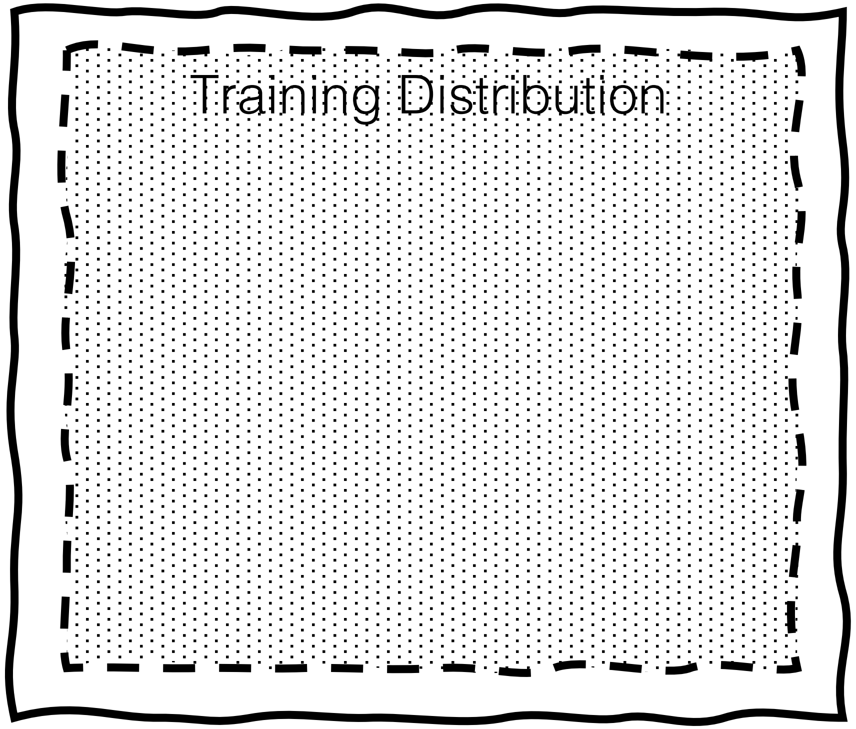
Real World Distribution



Embodied Data

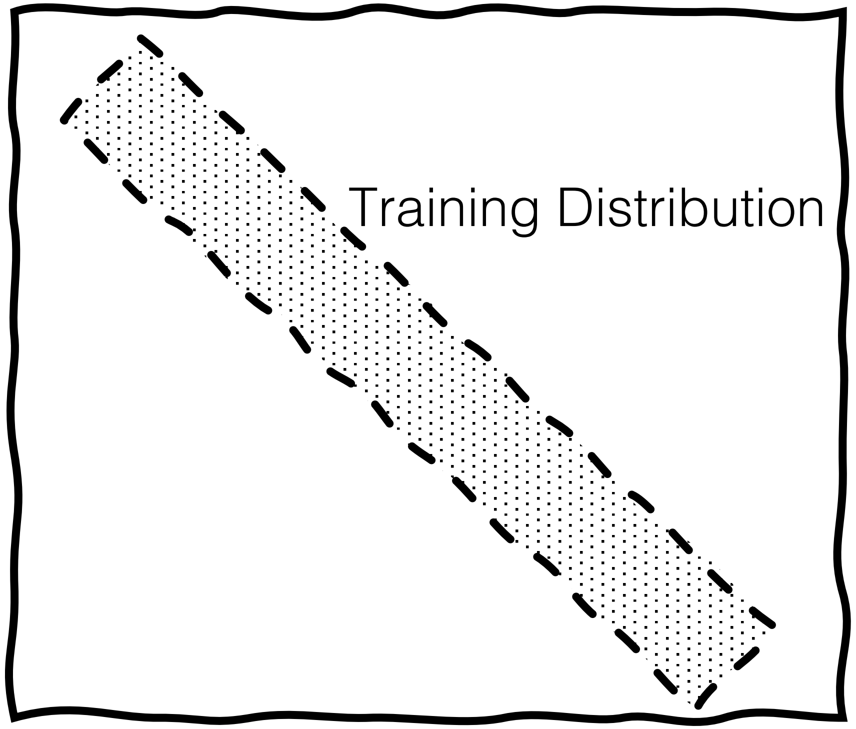
Why Does Generative AI on Other Settings Work Poorly?

Real World Distribution



Natural Language

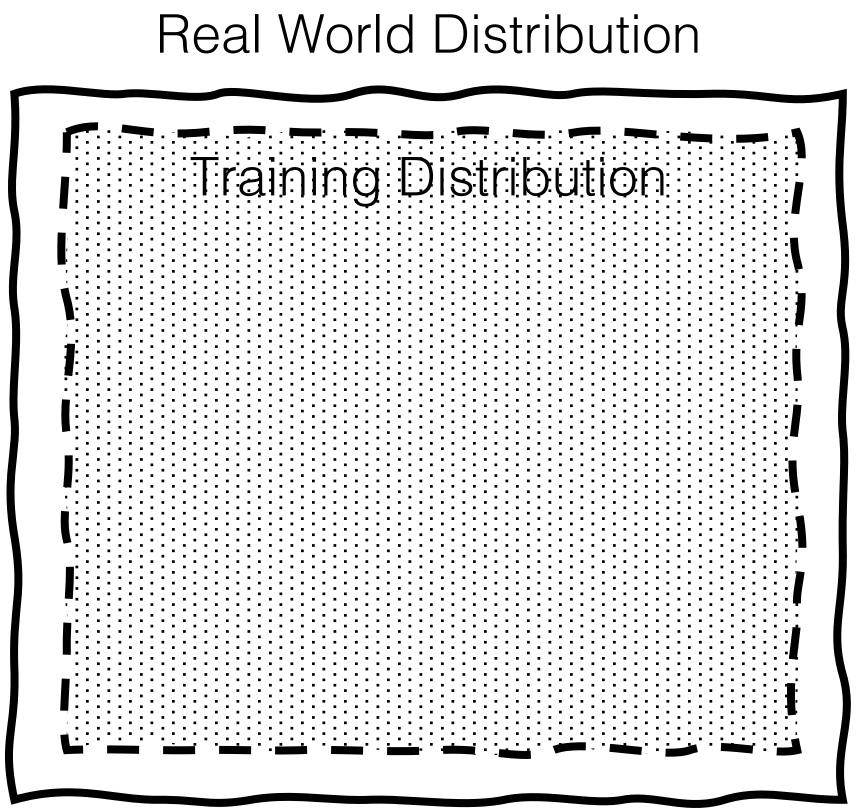
Real World Distribution



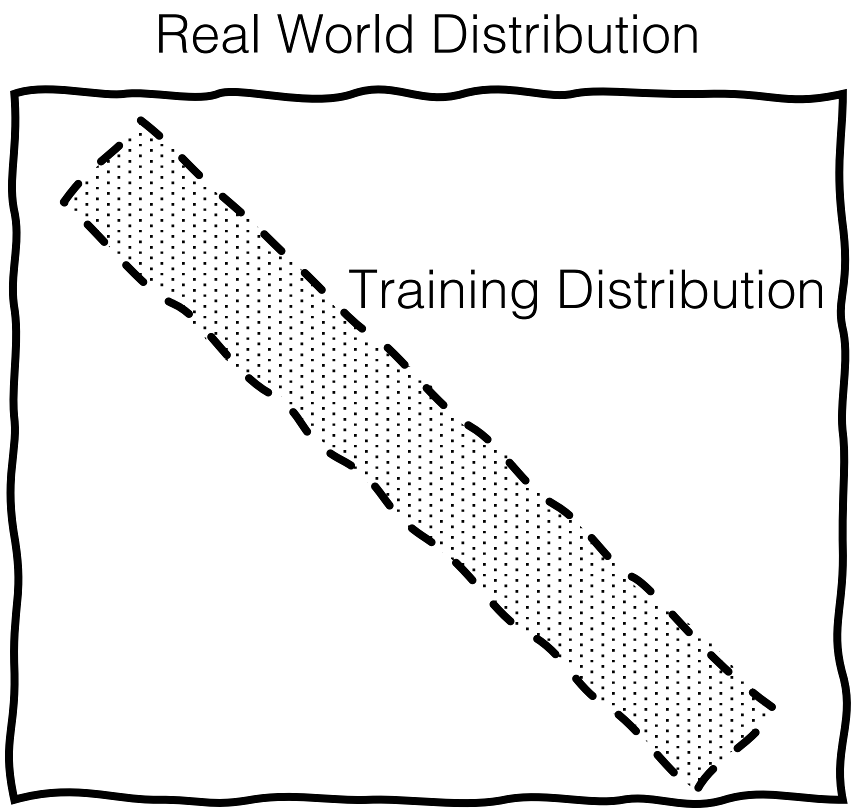
Embodied Data

Compared to pixels in images (and many other continuous distributions), natural language is much simpler (structured for effective communication) and naturally compositional (infinite use of finite means).

Composition of Learned Factors Yields Strong Generalization



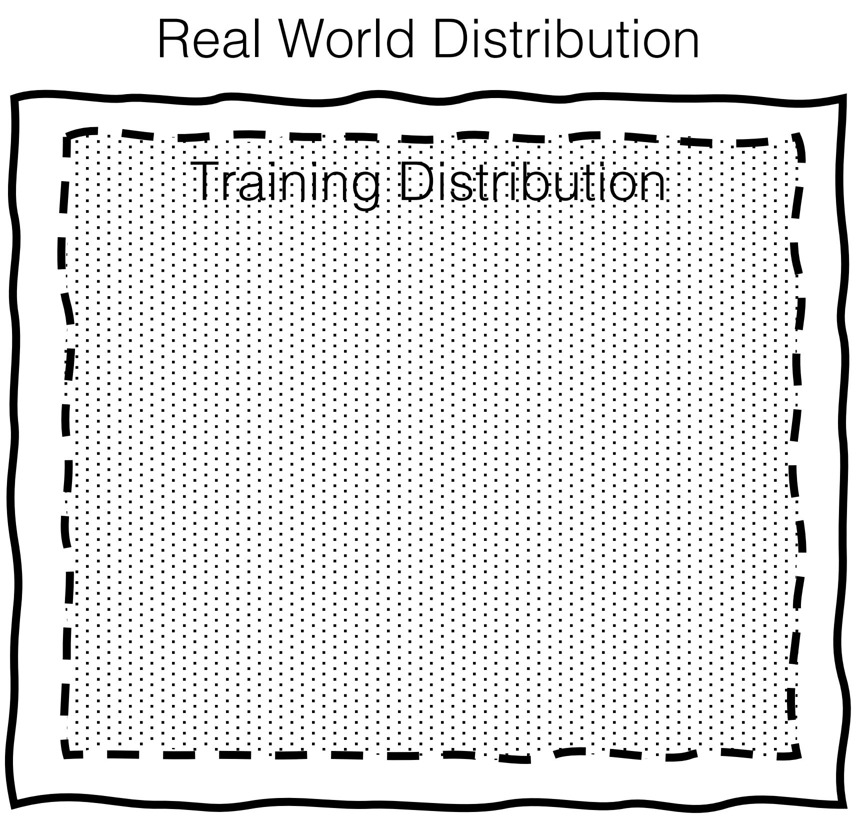
Natural Language



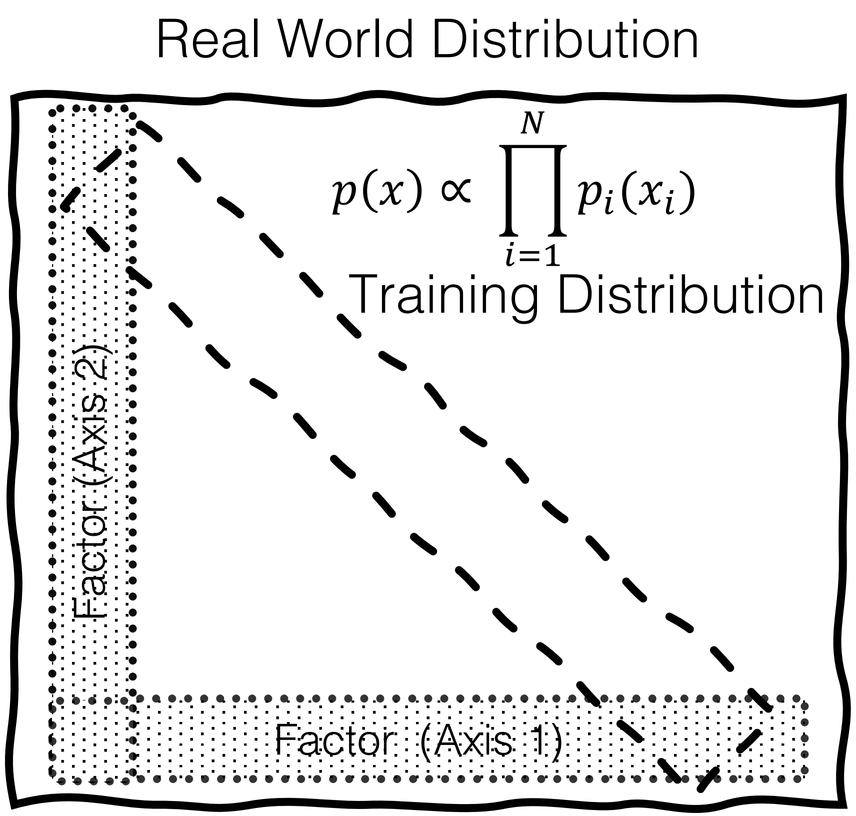
Embodied Data

Energy Based Models (EBMs) provide a probabilistic manner to represent the real distribution as a composition of factors!

Composition of Learned Factors Yields Strong Generalization



Natural Language

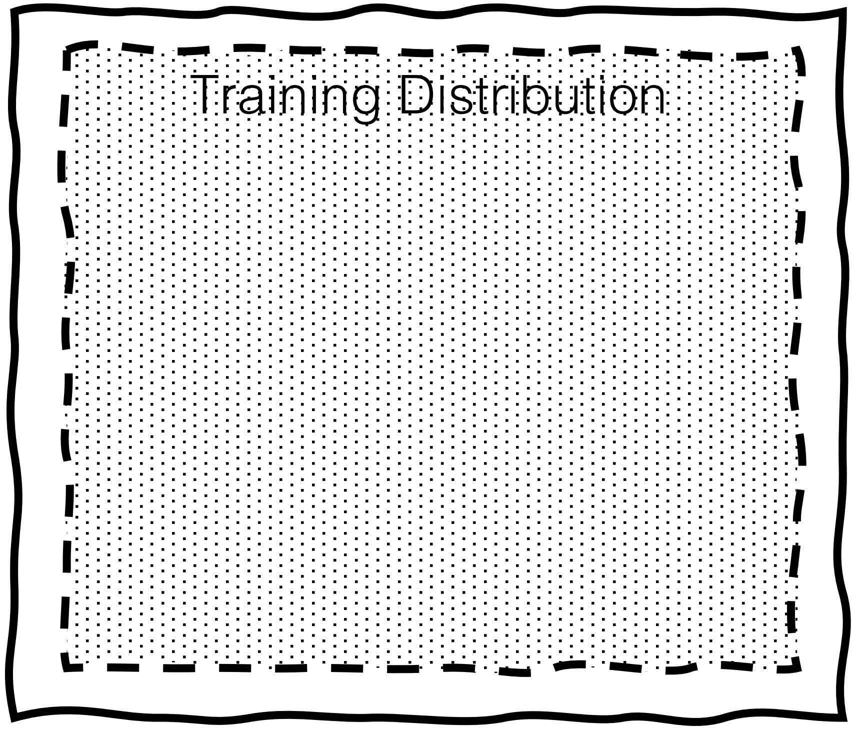


Embodied Data

Energy Based Models (EBMs) provide a probabilistic manner to represent the real distribution as a composition of factors!

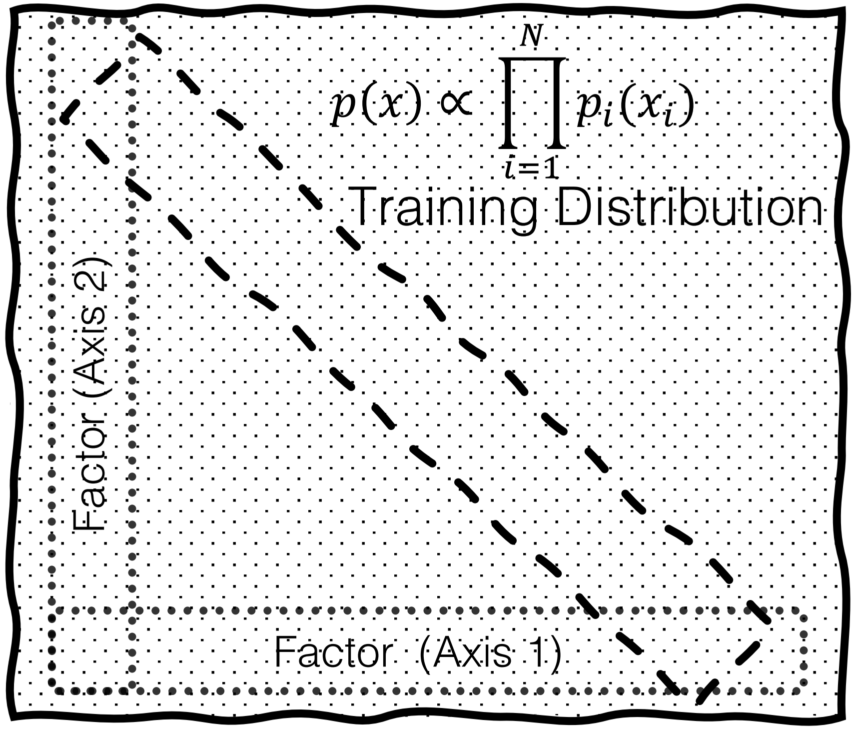
Composition of Learned Factors Yields Strong Generalization

Real World Distribution



Natural Language

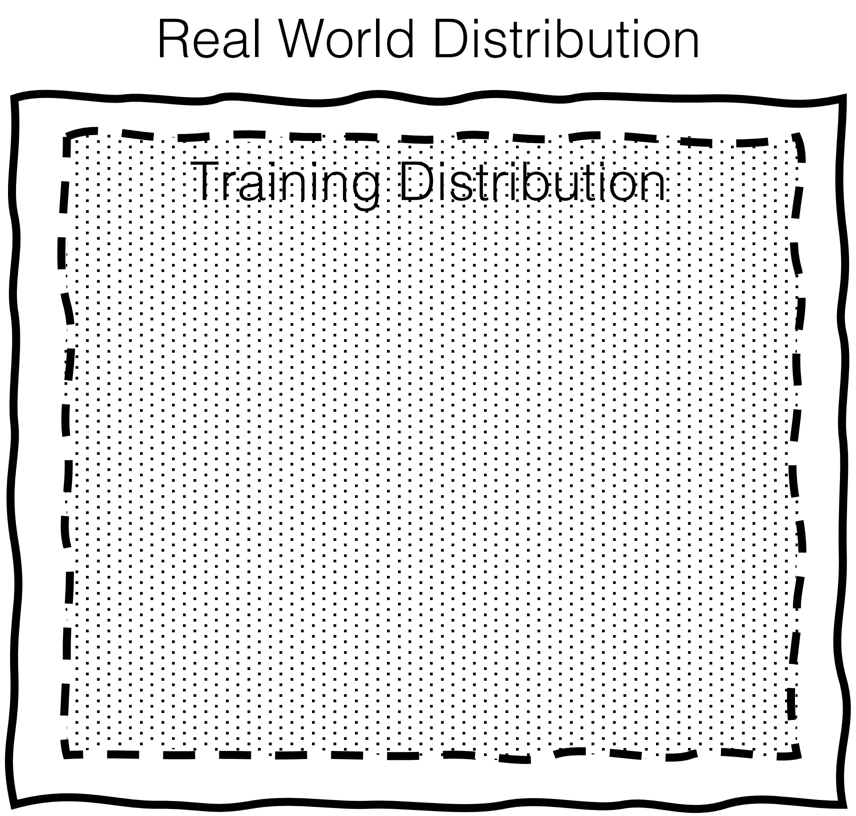
Real World Distribution



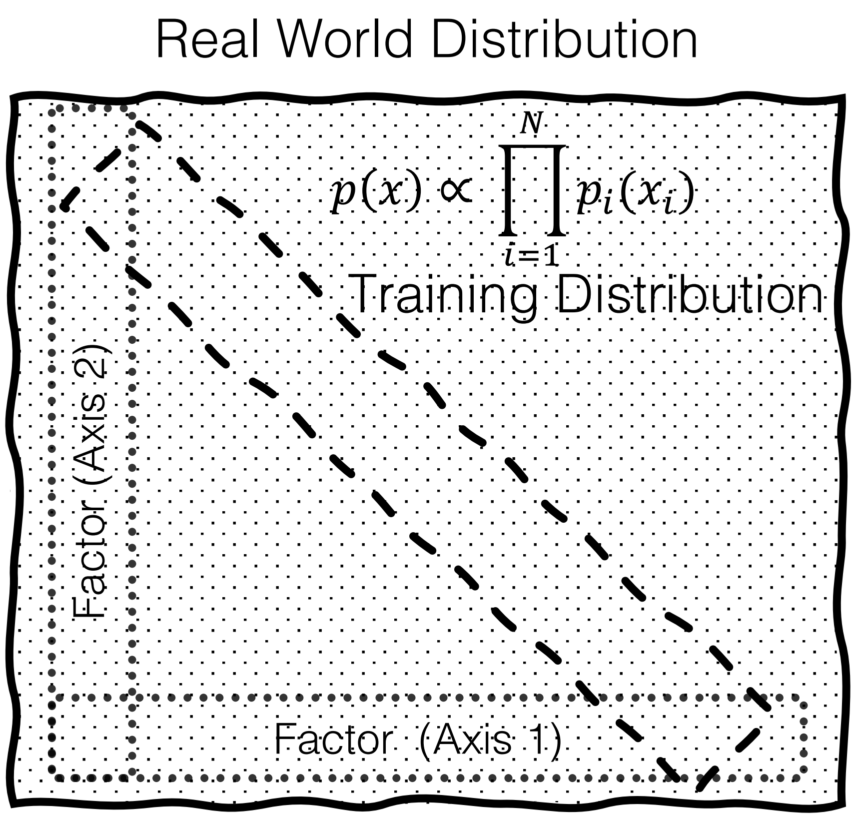
Embodied Data

Factors are combined to represent the entire distribution (even parts with no data).

Composition of Learned Factors Yields Strong Generalization



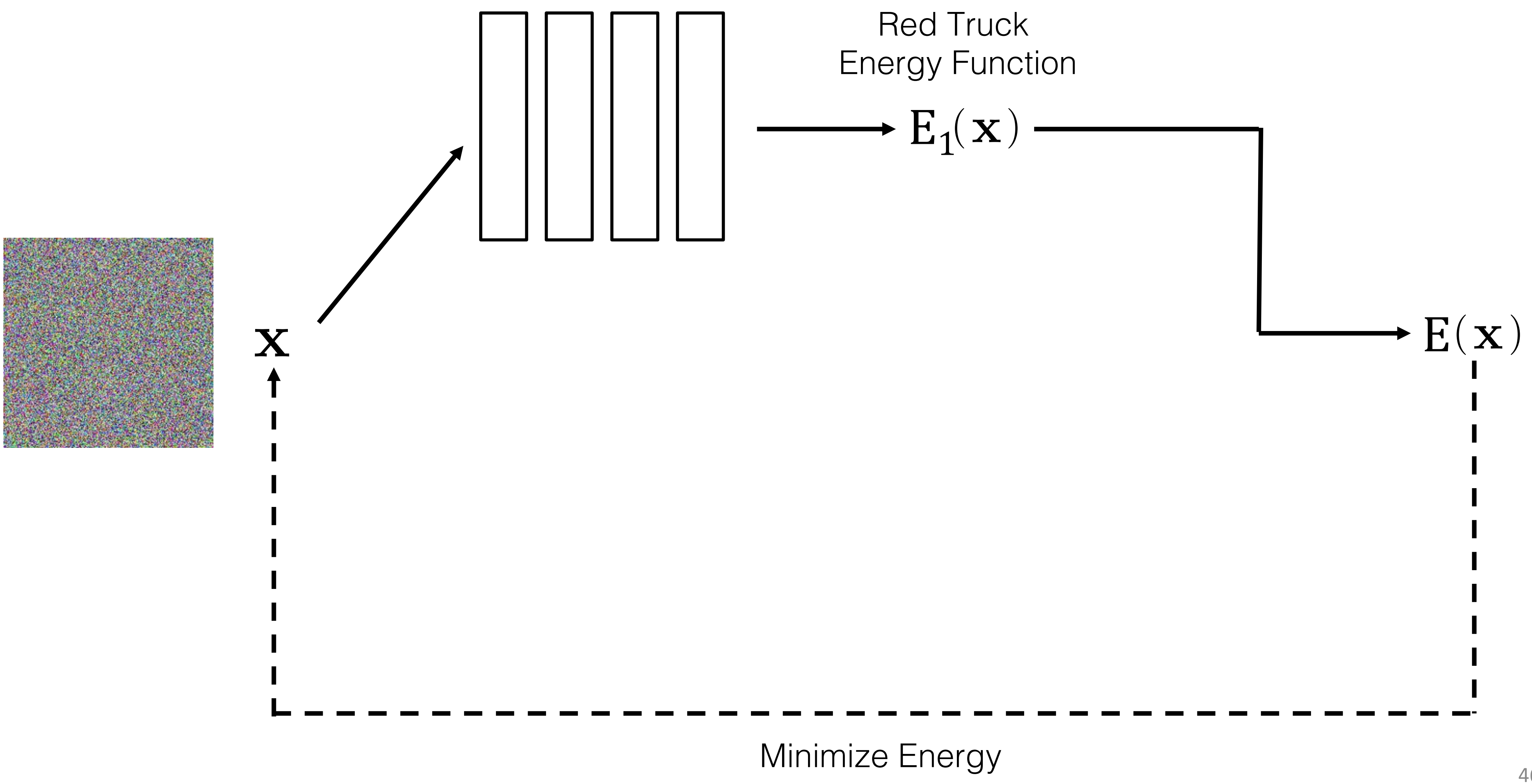
Natural Language



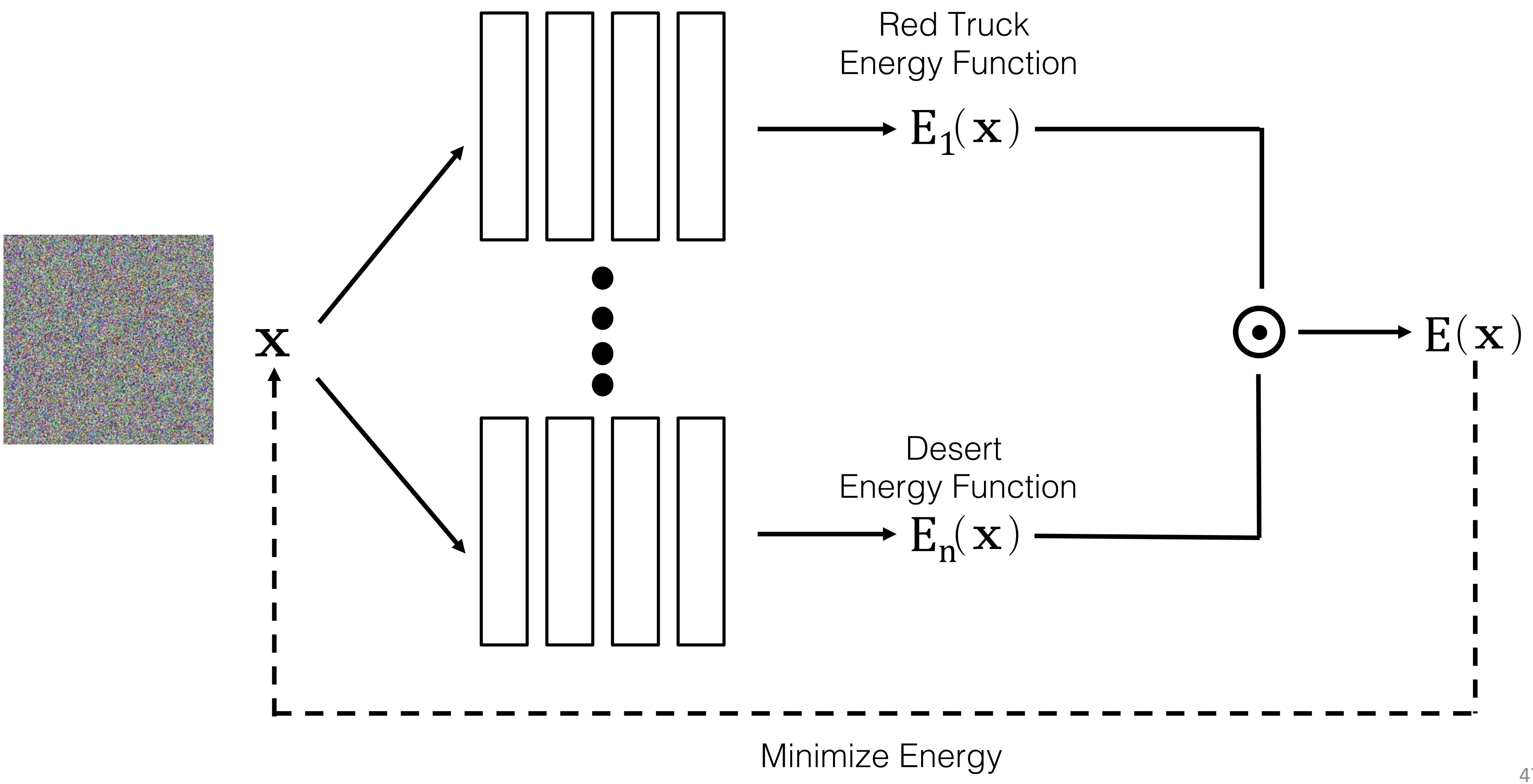
Embodied Data

Factors make independence assumptions about data that are biased.

Composing Different Energy Functions at Prediction Time

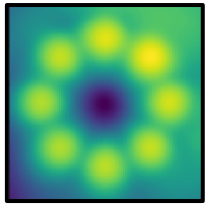


Composing Different Energy Functions at Prediction Time



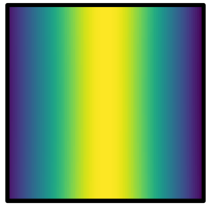
Combining Probability Distributions with Energy Functions

Products:



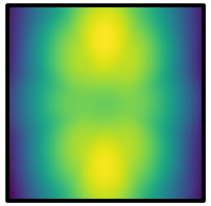
$p_1(x)$

$\times$



$p_2(x)$

=

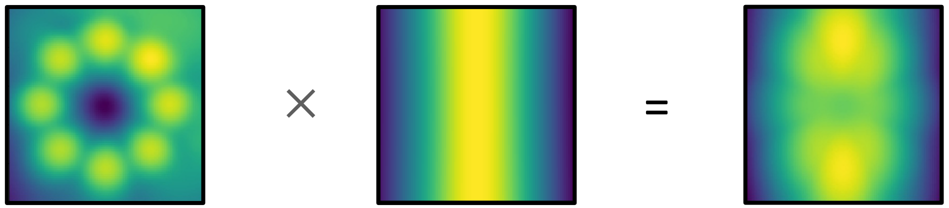


$p_1(x)\ p_2(x)$

[1] Du et al. Reduce, Reuse, Recycle: Compositional Generation with Energy-Based Diffusion Models and MCMC. ICML 2023

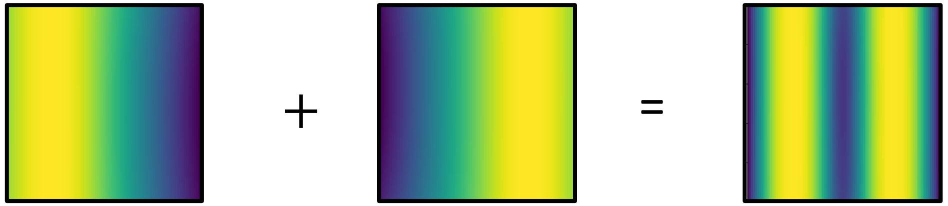
Combining Probability Distributions with Energy Functions

Products:



$p_1(x)$ $\times$ $p_2(x)$ $=$ $p_1(x)\ p_2(x)$

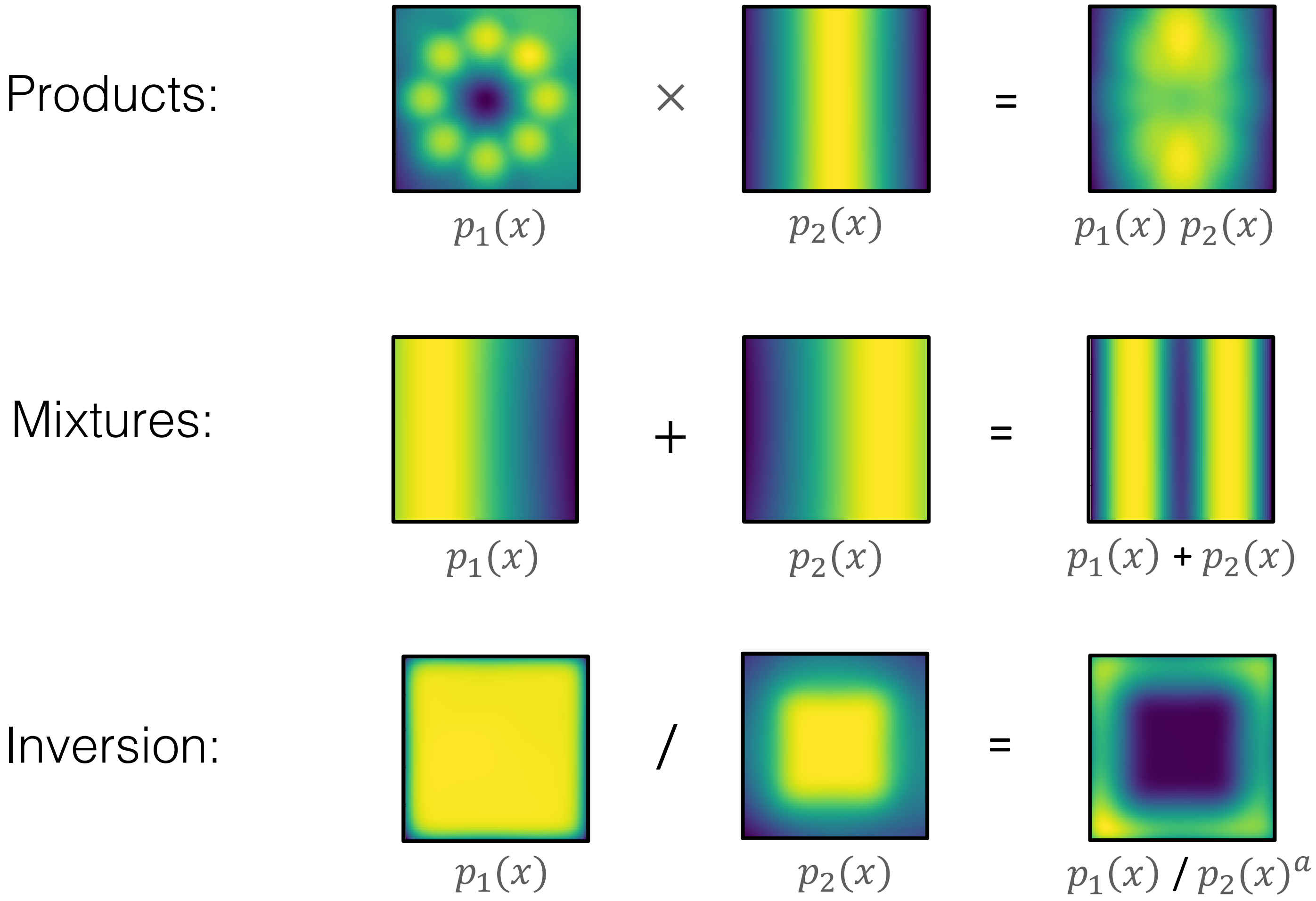
Mixtures:



$p_1(x)$ $+$ $p_2(x)$ $=$ $p_1(x) + p_2(x)$

[1] Du et al. Reduce, Reuse, Recycle: Compositional Generation with Energy-Based Diffusion Models and MCMC. ICML 2023

Combining Probability Distributions with Energy Functions

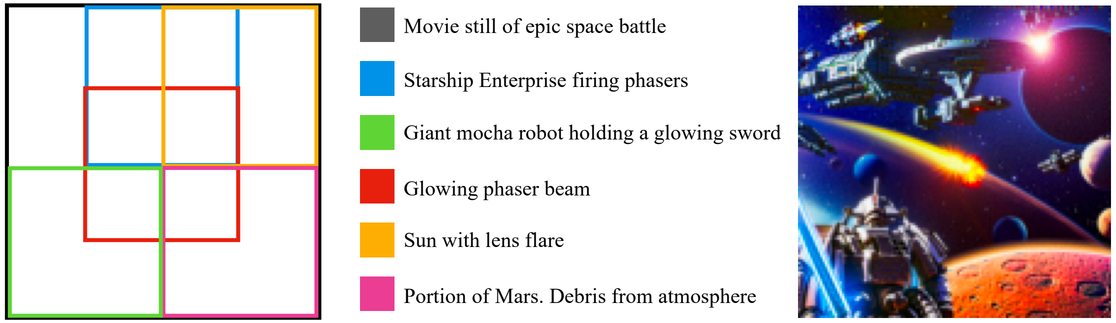


[1] Du et al. Reduce, Reuse, Recycle: Compositional Generation with Energy-Based Diffusion Models and MCMC. ICML 2023

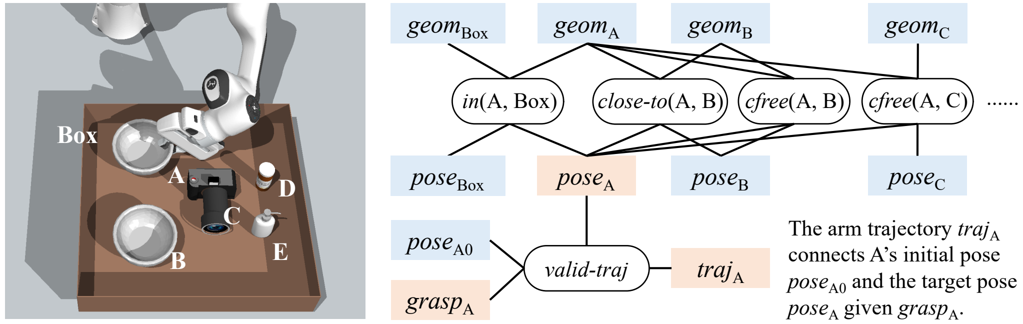
Applications of Composing Generative Models

[1] Liu*, Li*, **Du*** et al. Composable Visual Generation with Diffusion Models. ECCV 2022

[2] Yang*, **Du*** et al. Probabilistic Adaptation of Text-to-Video Models. ICLR 2024



[3] **Du** et al. Reduce, Reuse, Recycle: Compositional Generation with Energy-Based Diffusion Models and MCMC. ICML 2023

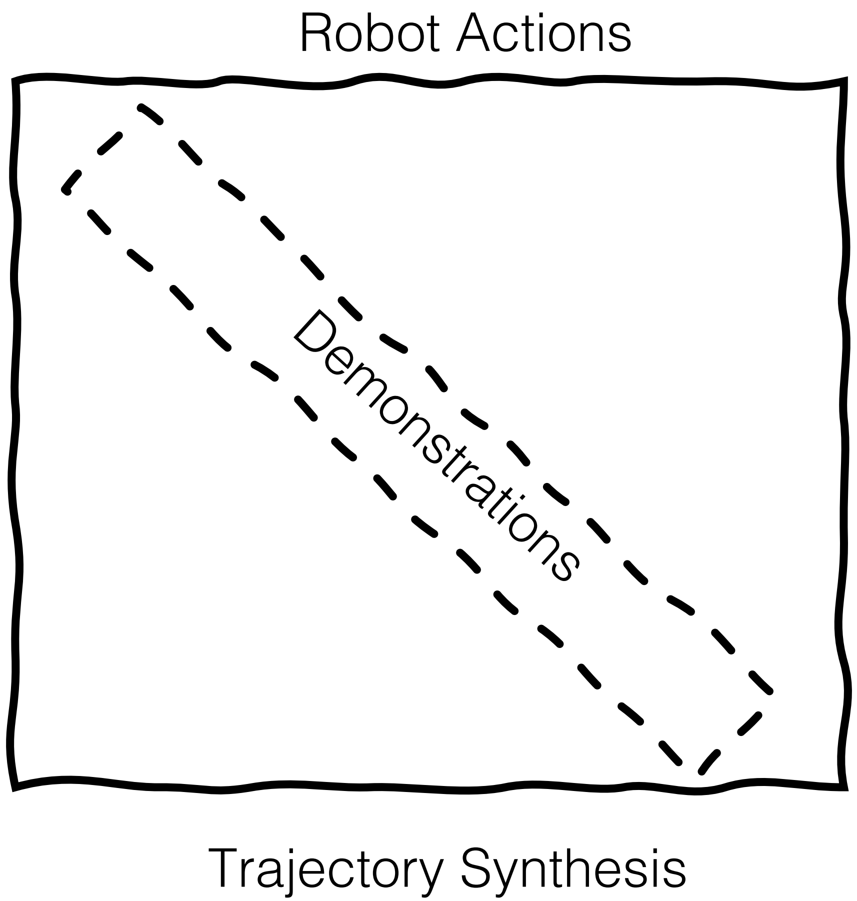


(a) Visualization of the environment while placing object A.

(b) Visualization of the constraint graphs associated with the object placement. There are three decision variables.

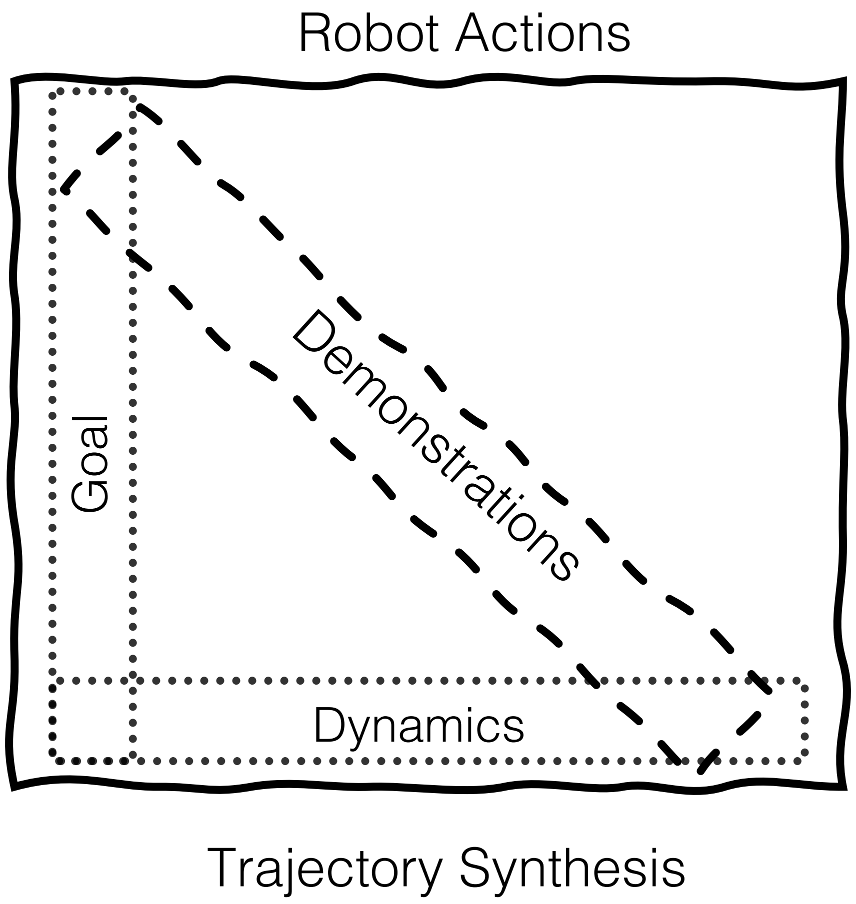
[4] Yang, Mao, **Du** et al. Compositional Diffusion-Based Continuous Constraint Solvers. CoRL 2023

Generalizing Beyond Demonstrations through Compoistion



We want to construct robotic agents that can generalize beyond the demonstrations they have seen!

Planning through Compositional Generation

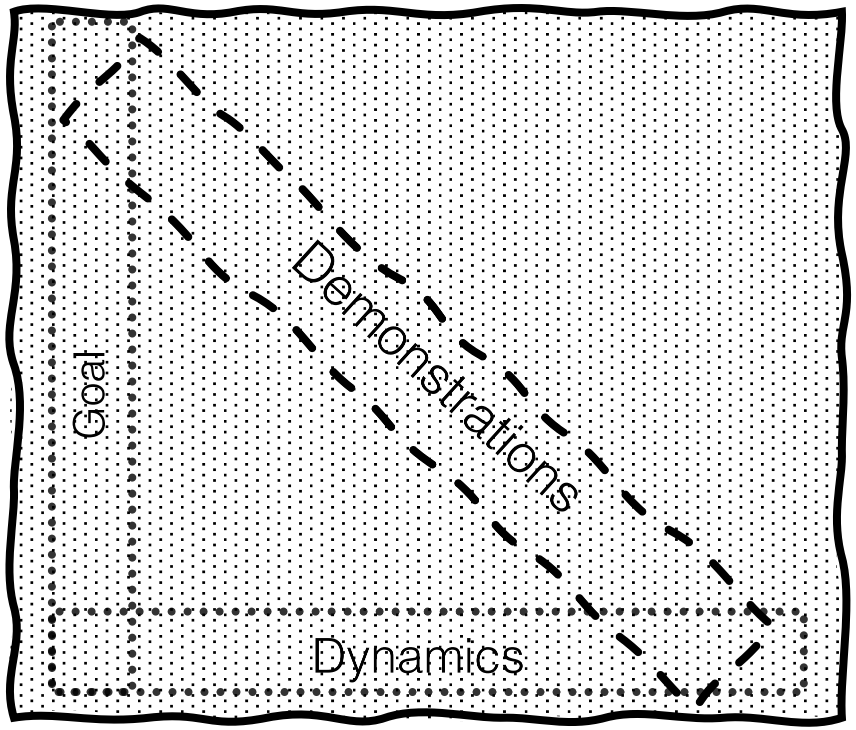


$$p_{\text{plan}}(\tau) \propto p_{\text{dynamics}}(\tau) p_{\text{task}}(\tau, \text{goal})$$

Decompose demonstrations into a model capturing the dynamics of the environment + a model to capture the goal of a task.

Planning through Compositional Generation

Robot Actions



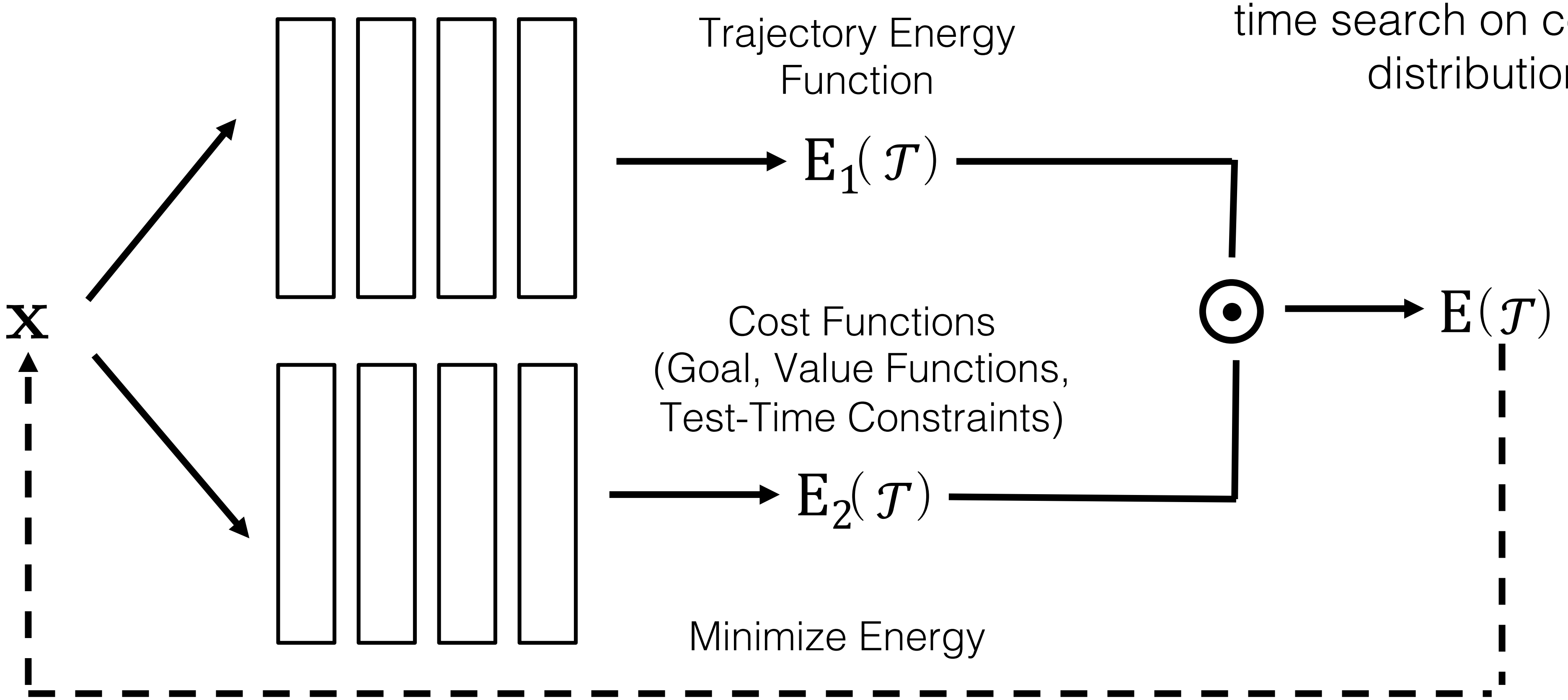
Trajectory Synthesis

$$p_{\text{plan}}(\tau) \propto p_{\text{dynamics}}(\tau) p_{\text{task}}(\tau, \text{goal})$$

Enables generalization to new combinations of states + goals through probabilistic planning!

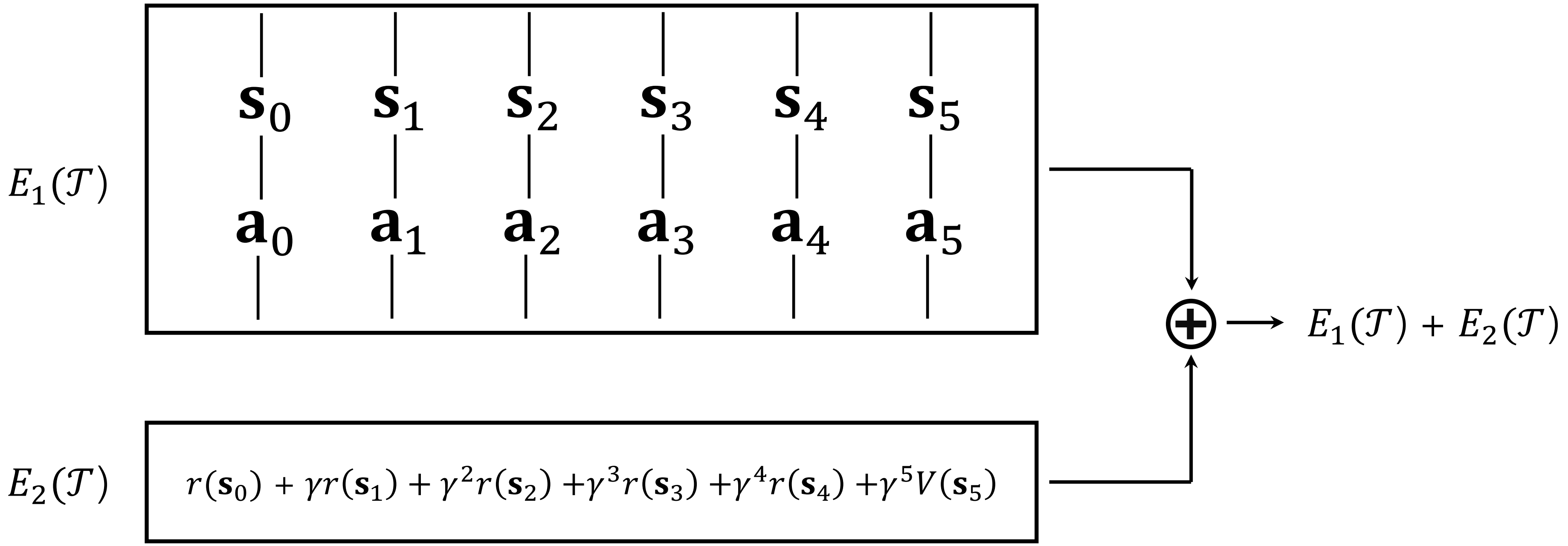
Planning with Energy Minimization

Planning/reinforcement learning through inference time search on composed distributions.

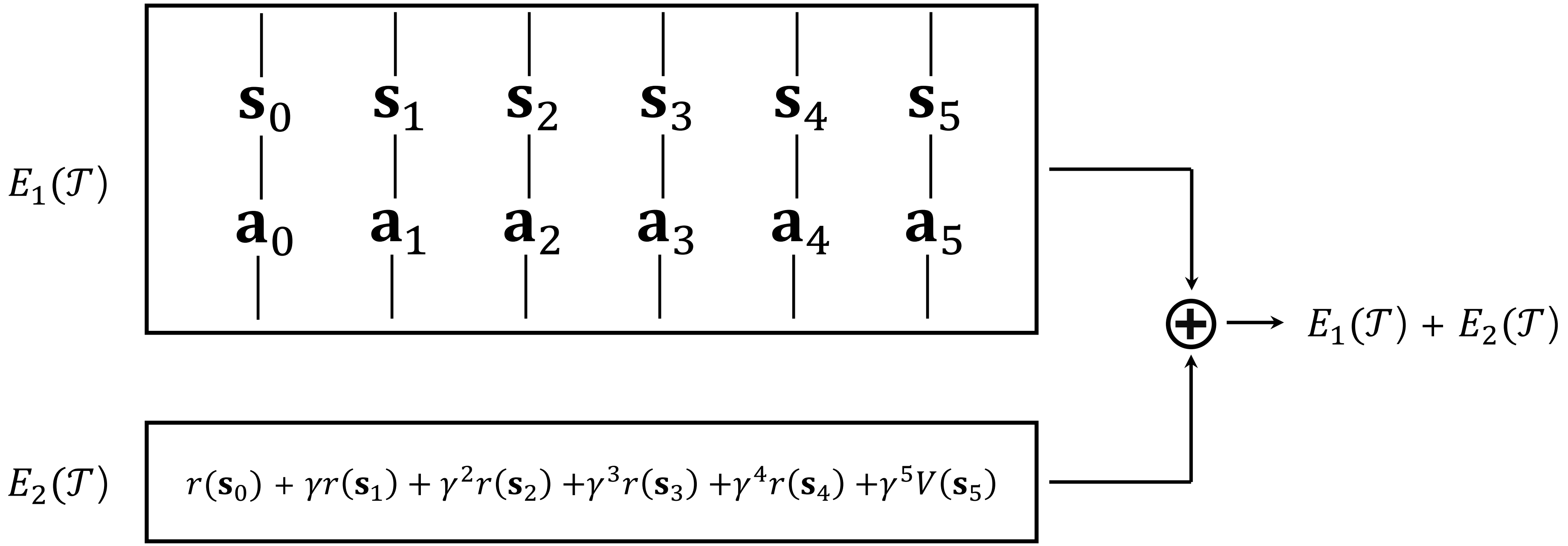


[1] **Du** et al. Model Based Planning with Energy Based Models. CoRL 2019.
[2] Janner*, **Du*** et al. Planning with Diffusion for Flexible Behavior Synthesis. ICML 2022
[3] Ajay*, **Du***, Gupta* et al. Is Conditional Generative Modeling all You Need For Decision Making? ICLR 2023

Reinforcement Learning through Value Composition

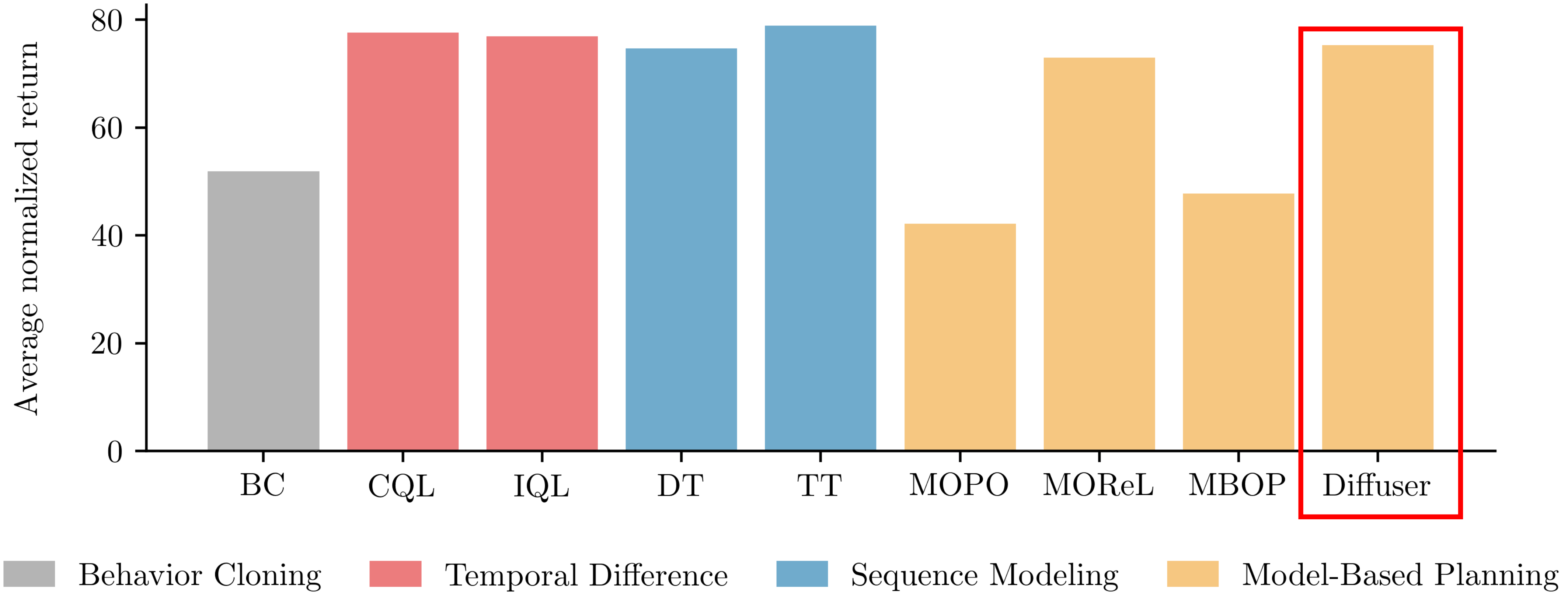


Reinforcement Learning through Value Composition

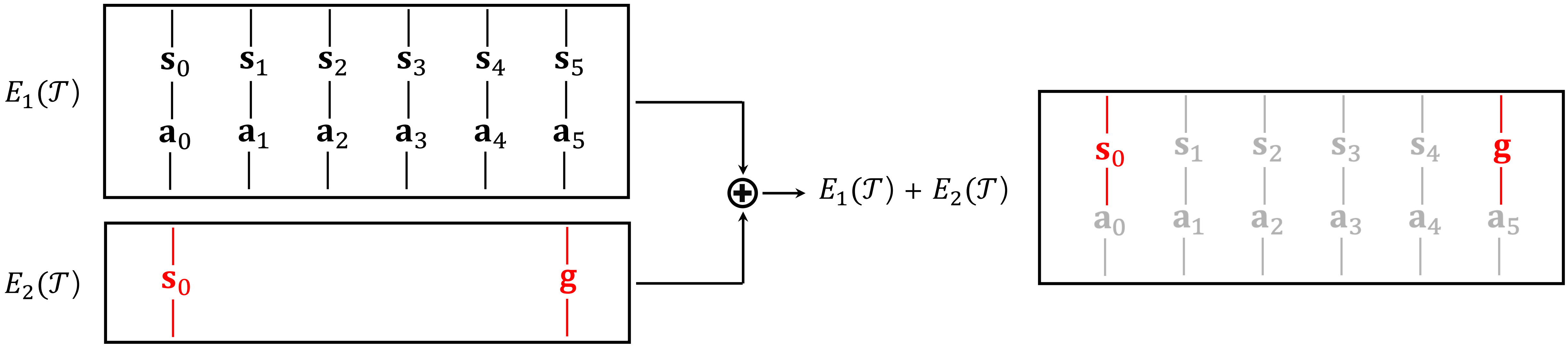


Solve many different tasks with one model!

Reinforcement Learning through Value Composition

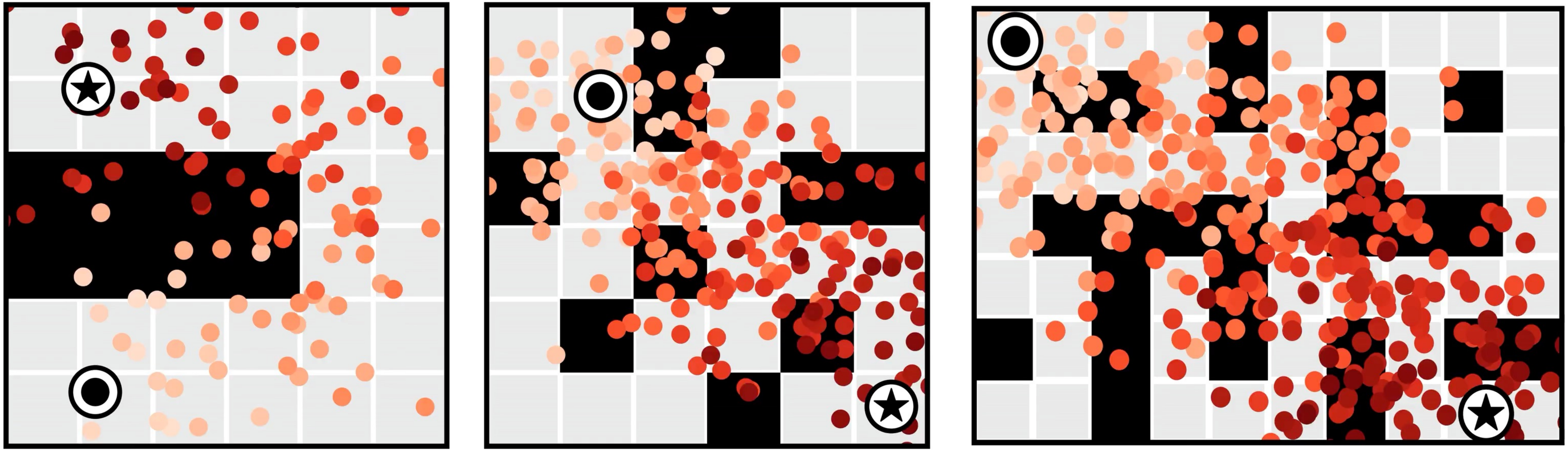


Goal Planning through Optimization

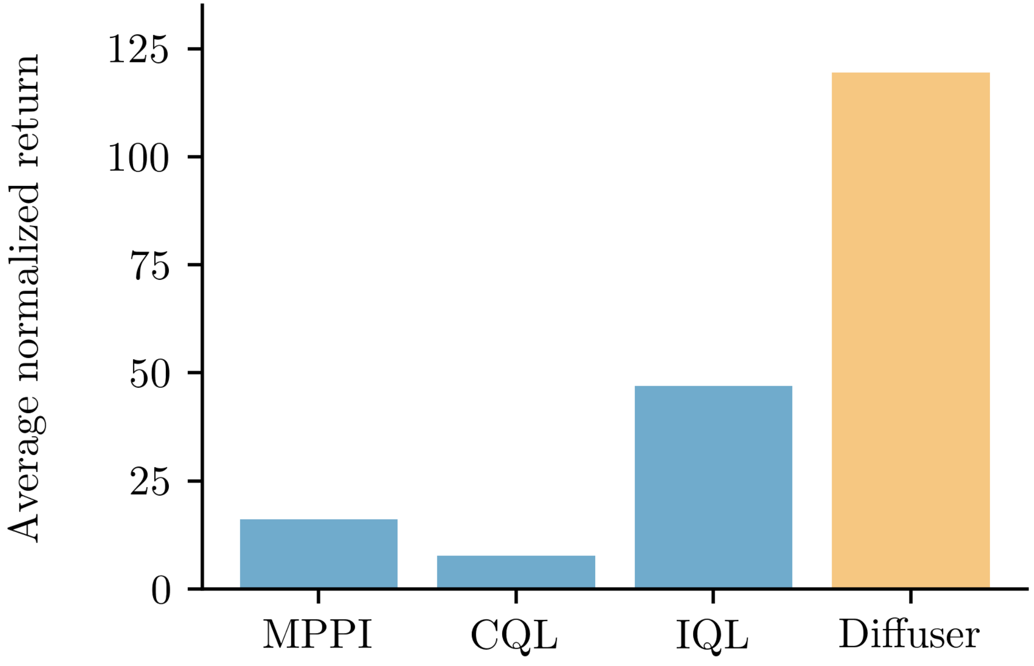


Construct a zero-shot goal-seeking policy!

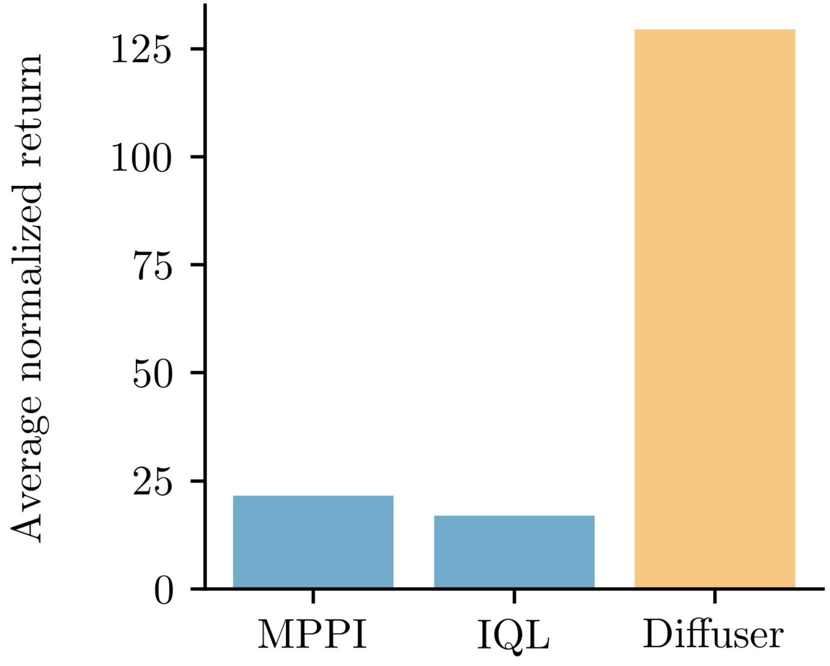
Goal Planning through Optimization



Goal Planning through Optimization



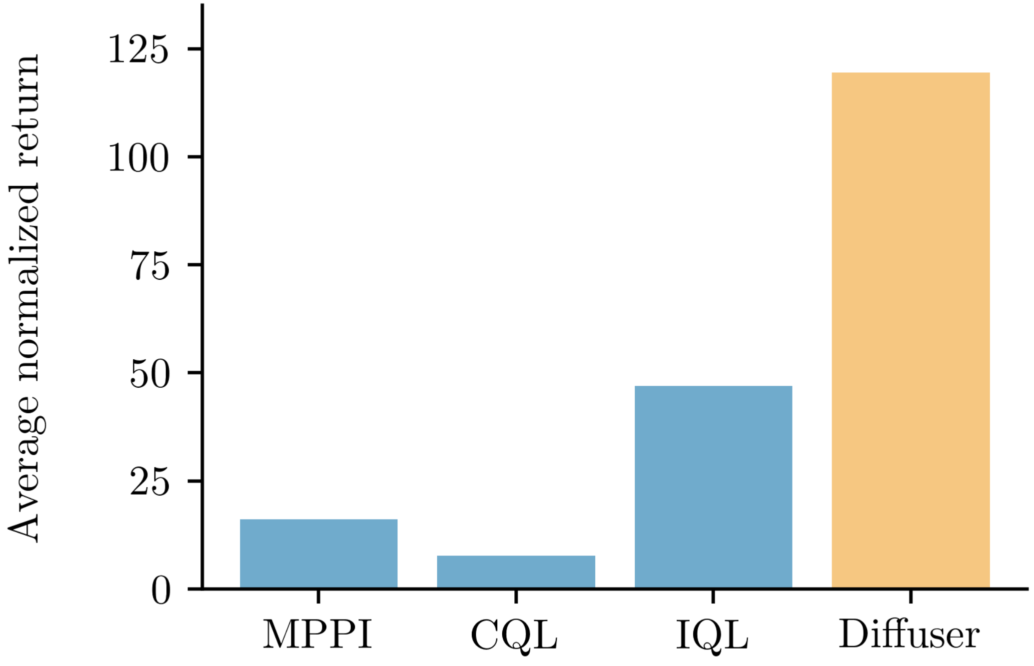
Single Task Planning



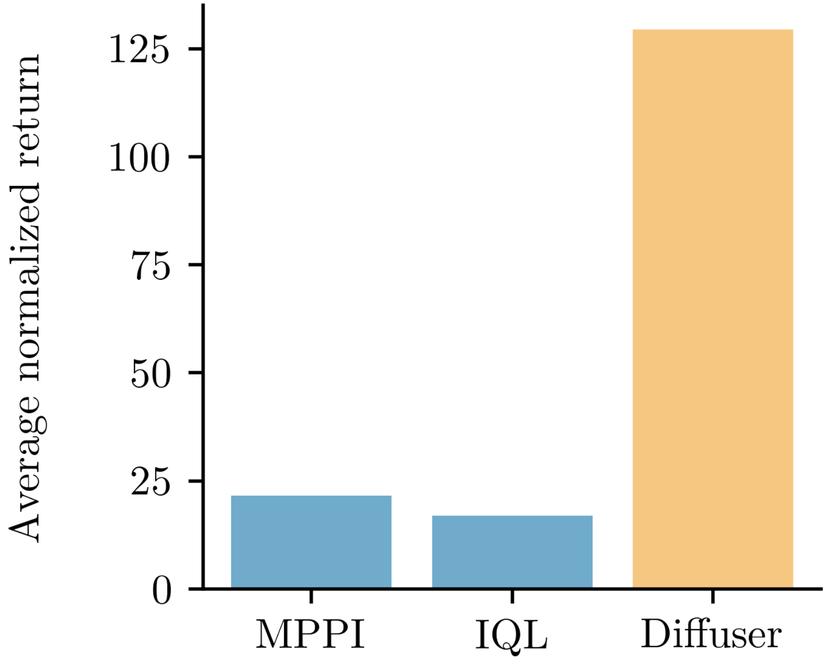
Multi-Task Planning

Goal Planning through Optimization

Baselines require per task retraining
while our trained model can be applied
across tasks!

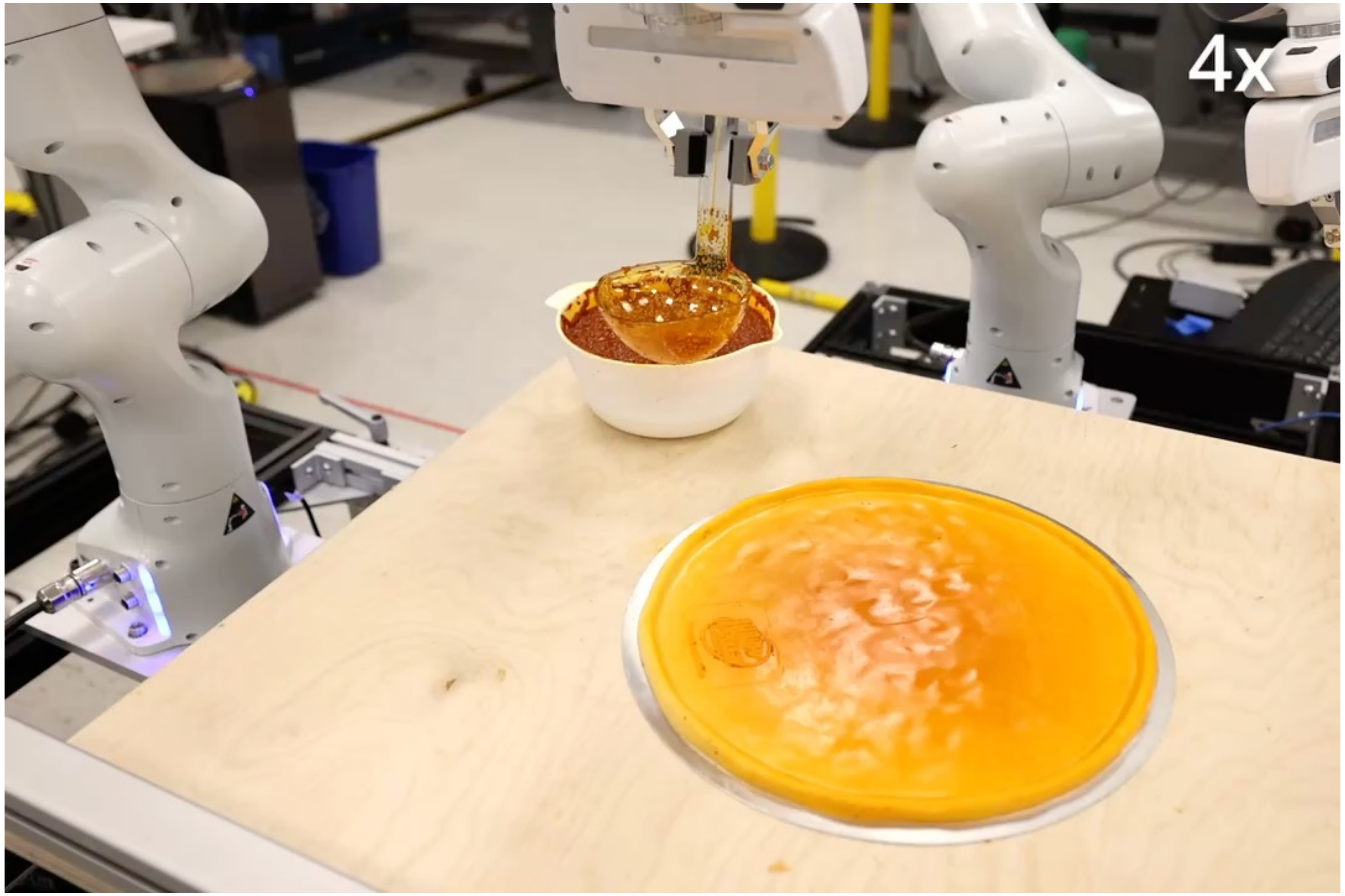


Single Task Planning



Multi-Task Planning

Planning From Partial Visual Observations on Real Robots



Can learn complex trajectory planning given only visual observations given very few (50) demos.

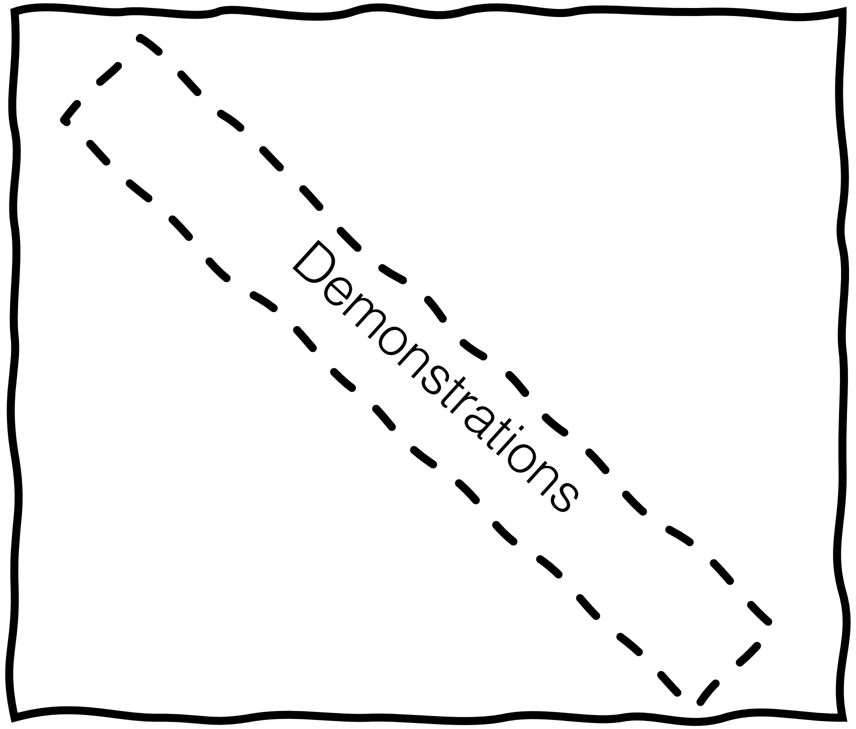
[1] Chi, Feng, **Du** et al. Diffusion Policy: Visualmotor Policy Learning via Action Diffusion. RSS 2023.

Solving Long Horizon Tasks by Composing Foundation Models

Make A Cup of Tea

Solving a long horizon decision-making task requires generalization across many sources of knowledge.

Embodied Agents



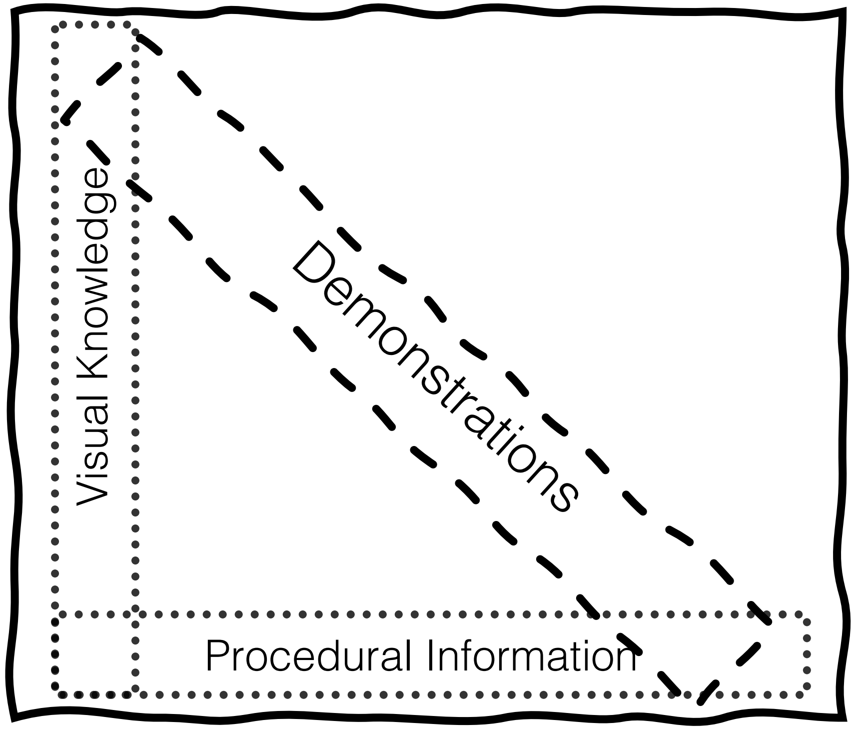
Decision Making

Composing Foundation Models for Embodied Agents

Make A Cup of Tea

Compose foundation models representing each axis of information!

Embodied Agents



Decision Making

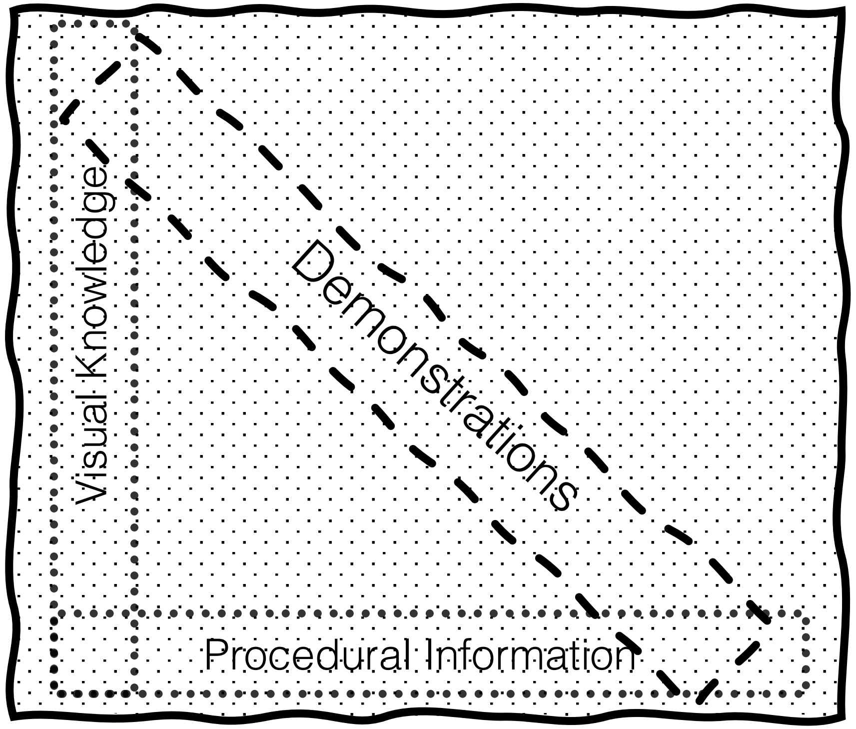
$$p_{\text{tea}}(\tau_{\text{text}}, \tau_{\text{image}}, \tau_{\text{action}}) \propto$$
$$p_{\text{LLM}}(\tau_{\text{text}})\, p_{\text{video}}(\tau_{\text{text}}, \tau_{\text{image}})\, p_{\text{action}}(\tau_{\text{image}}, \tau_{\text{action}})$$

Composing Foundation Models for Embodied Agents

Make A Cup of Tea

Compose foundation models representing each axis of information!

Embodied Agents



Decision Making

$$p_{\text{tea}}(\tau_{\text{text}}, \tau_{\text{image}}, \tau_{\text{action}}) \propto$$
$$p_{\text{LLM}}(\tau_{\text{text}})\, p_{\text{video}}(\tau_{\text{text}}, \tau_{\text{image}})\, p_{\text{action}}(\tau_{\text{image}}, \tau_{\text{action}})$$

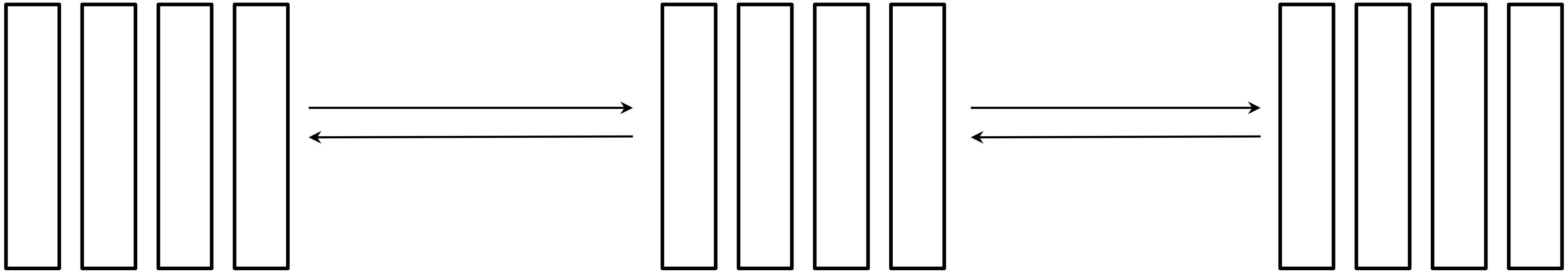
Hierarchical Planning with Foundation Models

1) Look for a tea kettle
2) Heat water
3) Find teabag
4) …

Task Information

Motion Information

Kinematics Information



Language Model

Video Model

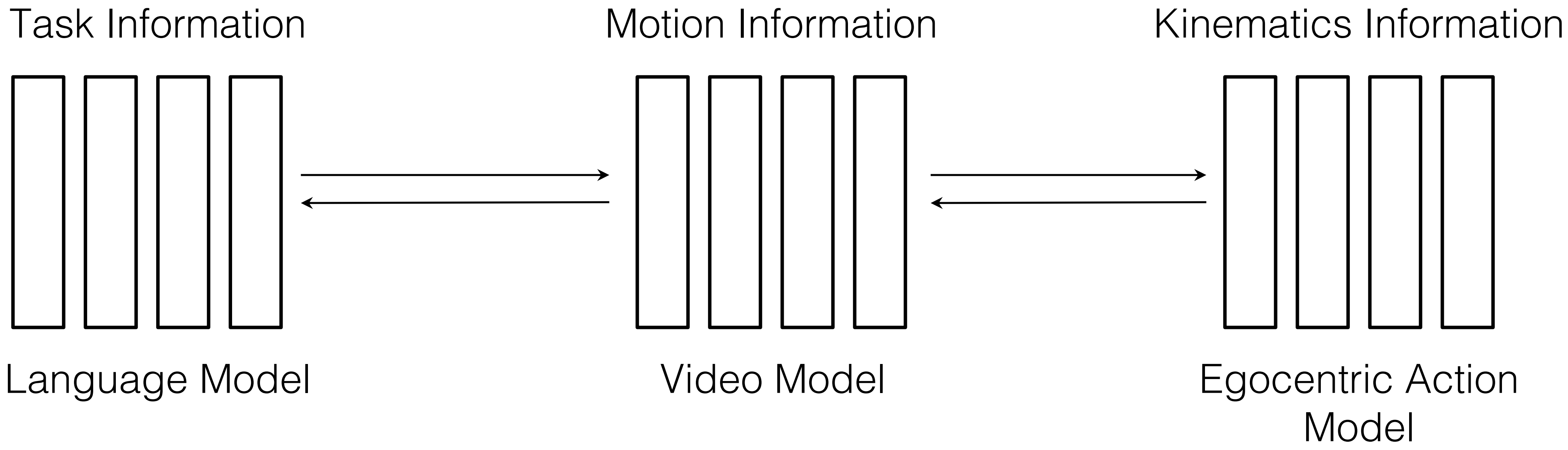
Egocentric Action Model

We want to construct a plan to make a cup of tea that is semantically, geometrically, and physically executable on a robot!

[1] Ajay*, Han*, **Du*** et al. Compositional Foundation Models for Hierarchical Planning. NeurIPS 2023

Hierarchical Planning with Foundation Models

$$p_{\mathrm{tea}}\left(\tau_{\mathrm{text}}, \tau_{\mathrm{image}}, \tau_{\mathrm{action}}\right) \propto$$
$$p_{\mathrm{LLM}}(\tau_{\mathrm{text}})\, p_{\mathrm{video}}\left(\tau_{\mathrm{text}}, \tau_{\mathrm{image}}\right) p_{\mathrm{action}}(\tau_{\mathrm{image}}, \tau_{\mathrm{action}})$$

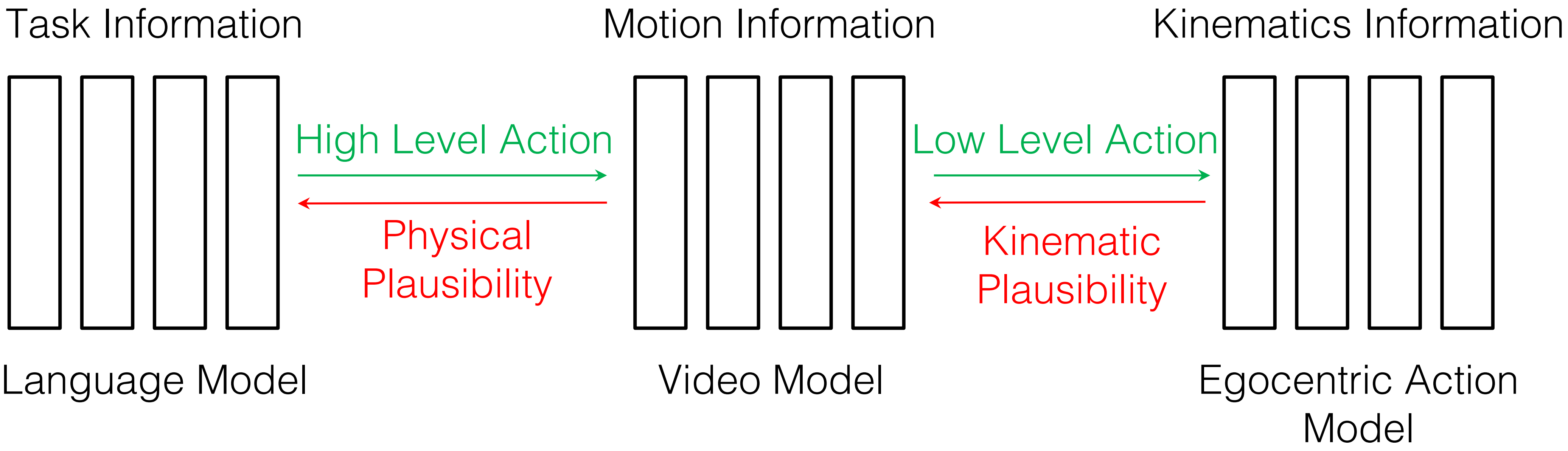


We want to construct a plan to make a cup of tea that is semantically, geometrically, and physically executable on a robot!

[1] Ajay*, Han*, **Du*** et al. Compositional Foundation Models for Hierarchical Planning. NeurIPS 2023

Hierarchical Planning with Foundation Models

$$p_{\text{tea}}(\tau_{\text{text}}, \tau_{\text{image}}, \tau_{\text{action}}) \propto$$
$$p_{\text{LLM}}(\tau_{\text{text}})\; p_{\text{video}}(\tau_{\text{text}}, \tau_{\text{image}})\; p_{\text{action}}(\tau_{\text{image}}, \tau_{\text{action}})$$



We want to construct a plan to make a cup of tea that is semantically, geometrically, and physically executable on a robot!

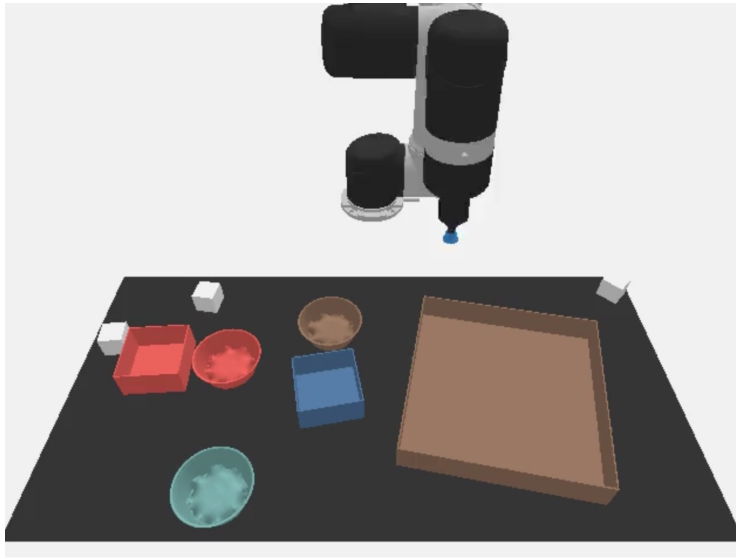
[1] Ajay*, Han*, **Du*** et al. Compositional Foundation Models for Hierarchical Planning. NeurIPS 2023

Hierarchical Decision Making with Multimodal Models

Goal

Stack red block on a cyan block and place a brown block to the right of stack.

Start Image



Generated Language Plan

Place cyan block in brown box.

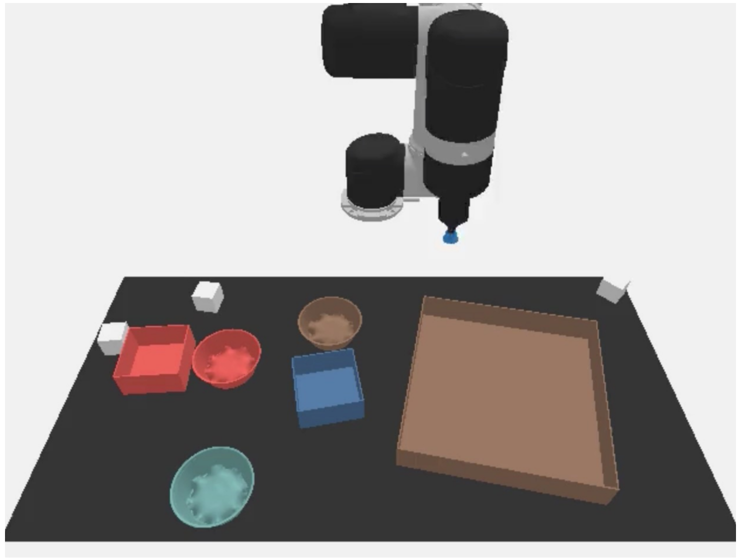
Generated Video Plan

Hierarchical Decision Making with Multimodal Models

Goal

Stack red block on a cyan block and place a brown block to the right of stack.

Start Image



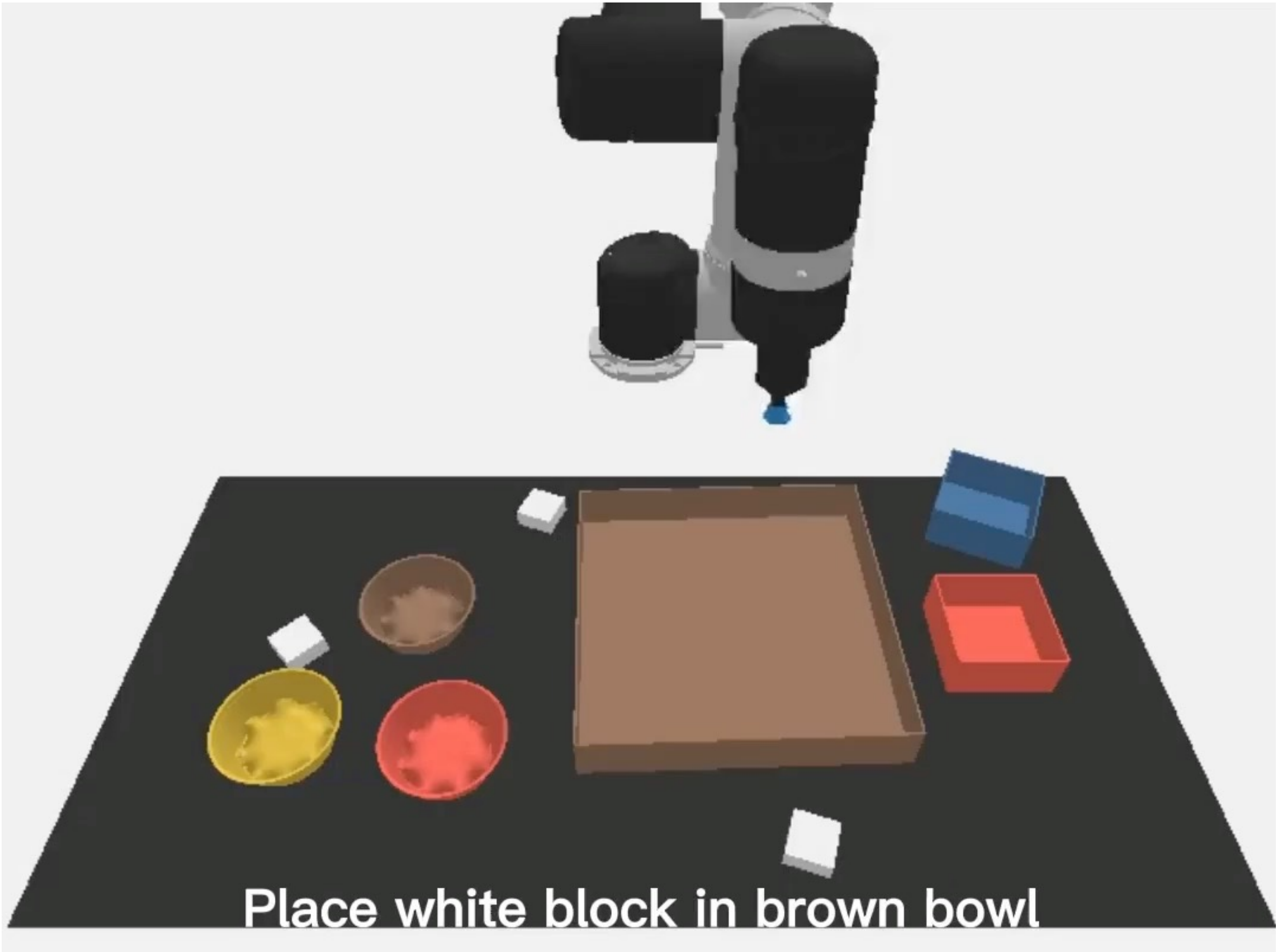
Generated Language Plan

Place white block in a cyan bowl.

Generated Video Plan

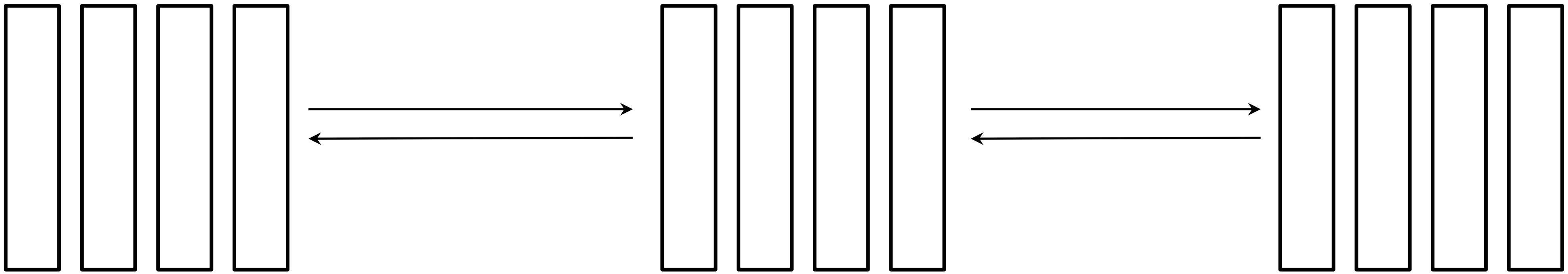
Hierarchical Decision Making with Multimodal Models Execution

Goal: Stack red block on top of brown block and place yellow block to the left of the stack



Zero-Shot Planning and Execution with Foundation Models

Task Information | Motion Information | Kinematics Information



Vision Language Model (PALM-E) | Video Model (UniPi) | Action Model (Goal-Conditioned RT2)

We can plan and execute unseen long horizon tasks on the real robot without any explicit task training!

[1] Du et al. Video Language Planning. ICLR 2024

Zero-Shot Planning and Execution with Foundation Models

Goal: Put the fruit into the top drawer.

Overview

- Introduction to Multimodal Generative Models
- Compositional Generative Models
- Discussion

Lecture 9: Generative AI, Part 1

Yilun Du

CS 2281R: Mathematical & Engineering Principles for Training Foundation Models