# Building Machines that Imagine and Reason

## Principles and Applications of Deep Generative Models


Shakir Mohamed

Google DeepMind

@shakir_za shakir@google.com



Deep Learning Summer School
August 2016

# Abstract


**Building Machines that Imagine and Reason:**
**Principles and Applications of Deep Generative Models**

Deep generative models provide a solution to the problem of unsupervised learning, in which a machine learning system is required to discover the structure hidden within unlabelled data streams. Because they are generative, such models can form a rich imagery the world in which they are used: an imagination that can harnessed to explore variations in data, to reason about the structure and behaviour of the world, and ultimately, for decision-making. This tutorial looks at how we can build machine learning systems with a capacity for imagination using deep generative models, the types of probabilistic reasoning that they make possible, and the ways in which they can be used for decision making and acting.

Deep generative models have widespread applications including those in density estimation, image denoising and in-painting, data compression, scene understanding, representation learning, 3D scene construction, semi-supervised classification, and hierarchical control, amongst many others. After exploring these applications, we'll sketch a landscape of generative models, drawing-out three groups of models: fully-observed models, transformation models, and latent variable models. Different models require different principles for inference and we'll explore the different options available. Different combinations of model and inference give rise to different algorithms, including auto-regressive distribution estimators, variational auto-encoders, and generative adversarial networks. Although we will emphasise deep generative models, and the latent-variable class in particular, the intention of the tutorial is to explore the general principles, tools and tricks that can be used throughout machine learning. These reusable topics include Bayesian deep learning, variational approximations, memoryless and amortised inference, and stochastic gradient estimation. We'll end by highlighting the topics that were not discussed, and imagine the future of generative models.

# Motivations for machine learning

**Statistical and mathematical foundations**

**New era of scientific discovery**

**Disrupt and create new markets**

**Quest to solve intelligence**

**What components form the ideal machine learning system?**

# Why Generative Models

Move beyond associating inputs to outputs

Understand and imagine how the world evolves

Recognise objects in the world and their factors of variation

Detect surprising events in the world

Establish concepts as useful for reasoning and decision making

Imagine and generate rich plans for the future

**Part of a suite of complementary learning systems**

$f_\theta(\cdot)$ **Functions are deep networks**

Fully-connected, convolutional, recurrent

**Some Themes**

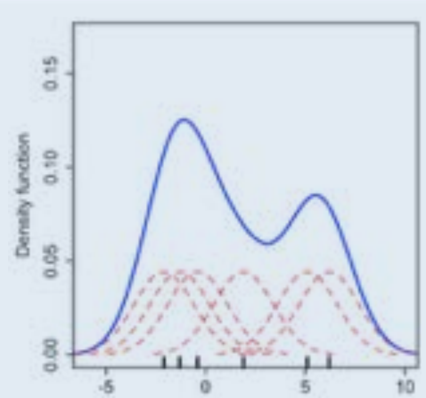
Design of probabilistic models

Bayesian Deep Learning

Memoryless and Amortised Inference

Stochastic Optimisation

Reasoning and Control



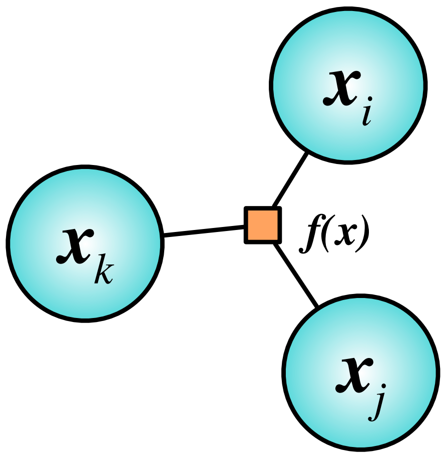

In some way, will involve the problem of **density estimation.**

**Part I**

# Landscape of Generative Models

Birds eye view of the current state of the art.

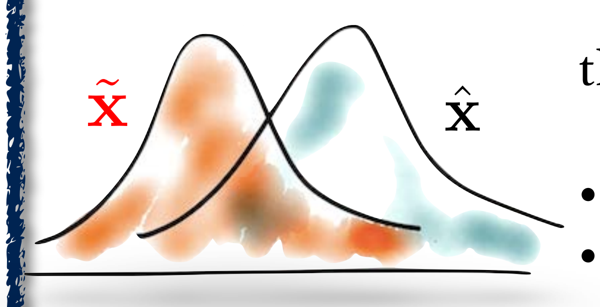
**Part II**




# A Model for Every Occasion

Explore three classes of generative models, their inductive biases, and implications for learning and algorithm design.

**Part III**

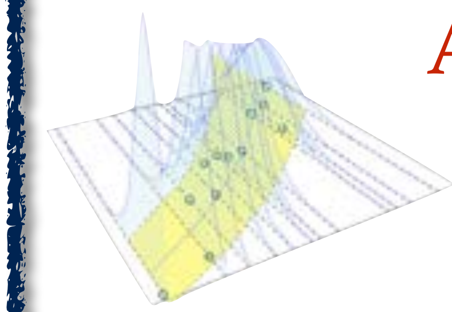



# Inference and Learning

Principles and approximations that can be used to drive learning in different types of models.

- Bayesian two-sample tests
- Marginal likelihood estimation

**Part IV**

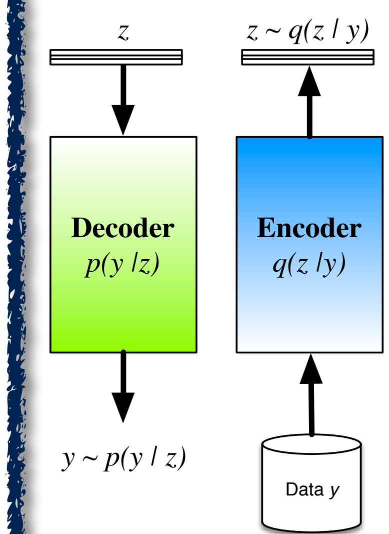


# Tools for Algorithm Building

Constructing scalable algorithms

- Stochastic approximation
- Amortised inference
- Stochastic optimisation

**Part V**




# The Case of Variational Auto-encoders

Explore different types of VAEs

- Discrete and continuous latent variables.
- Static, sequential, volumetric.
- Differentiable and non-differentiable fns.

**Part VI**

# Summary

Mention of things not discussed and wrap-up

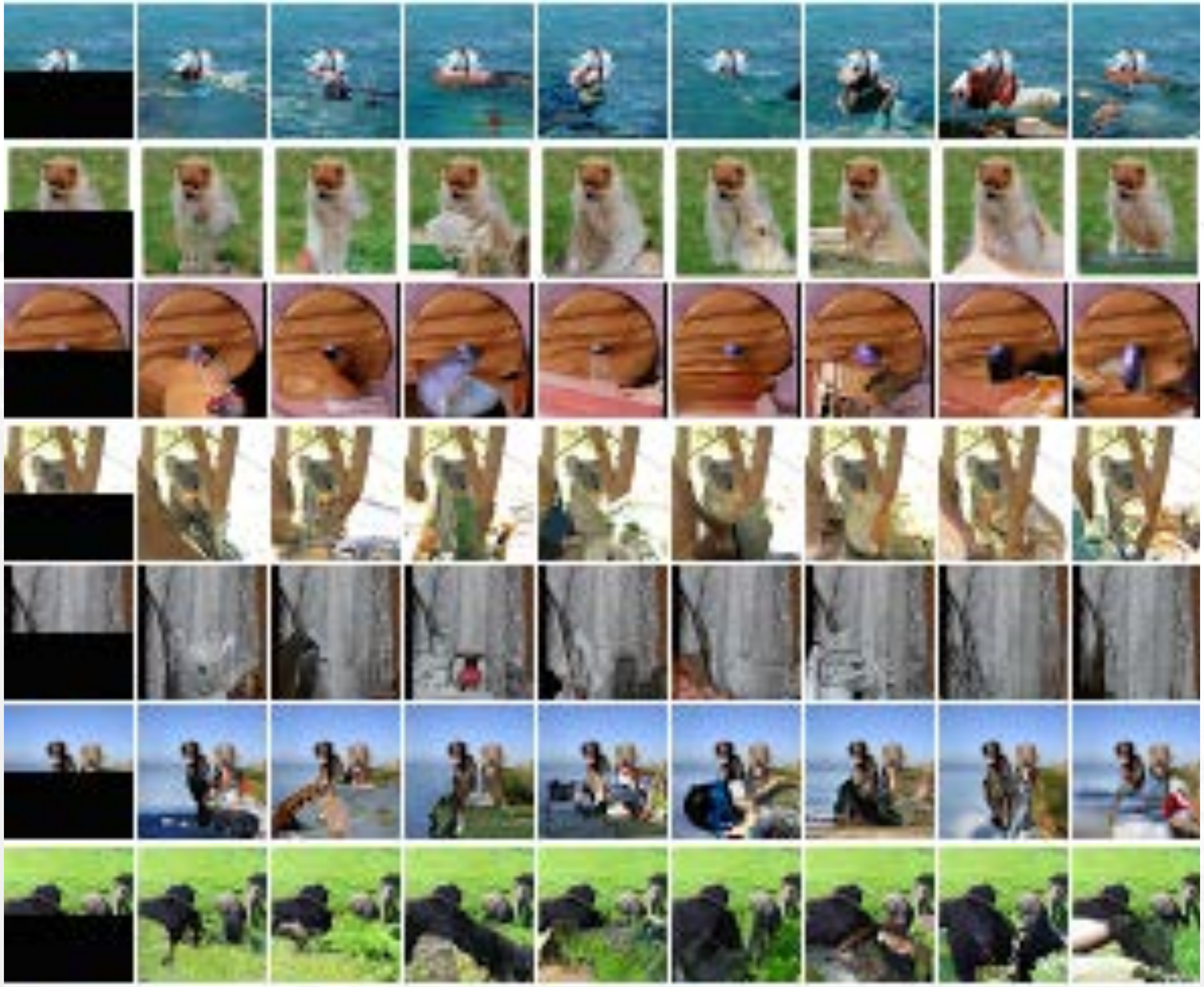
# Part I

# Landscape of Generative Models

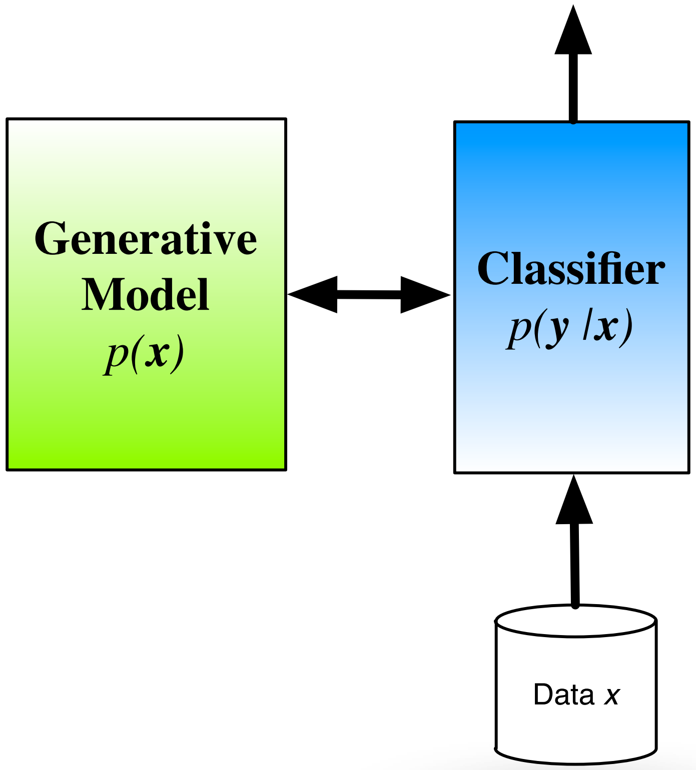
## Diversity of Applications and Progress

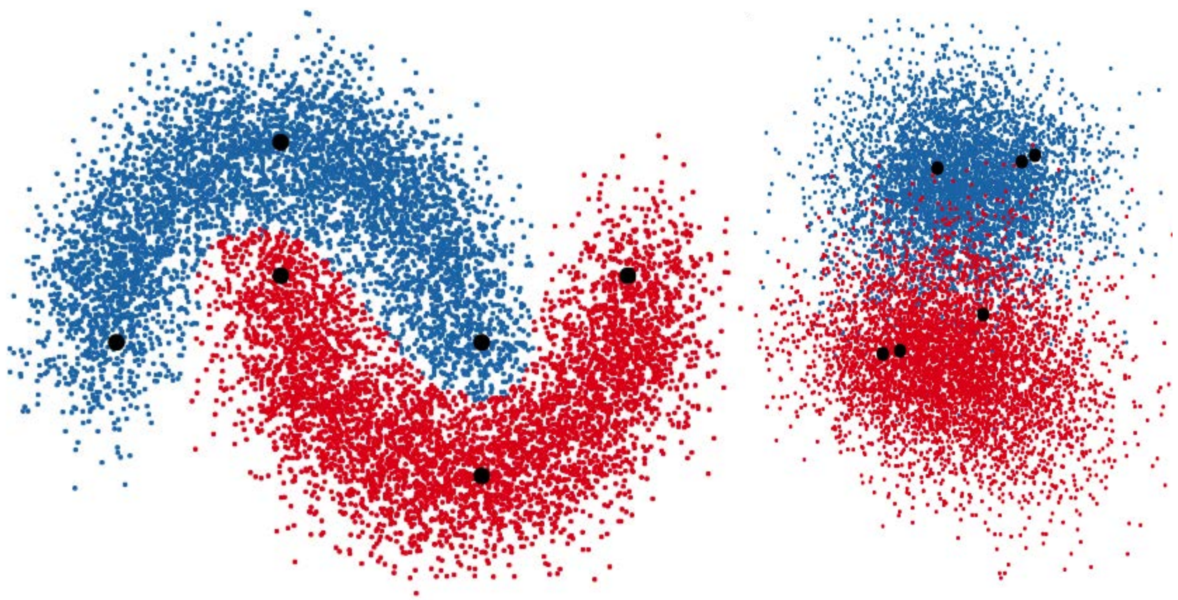
Fill
in
the
______

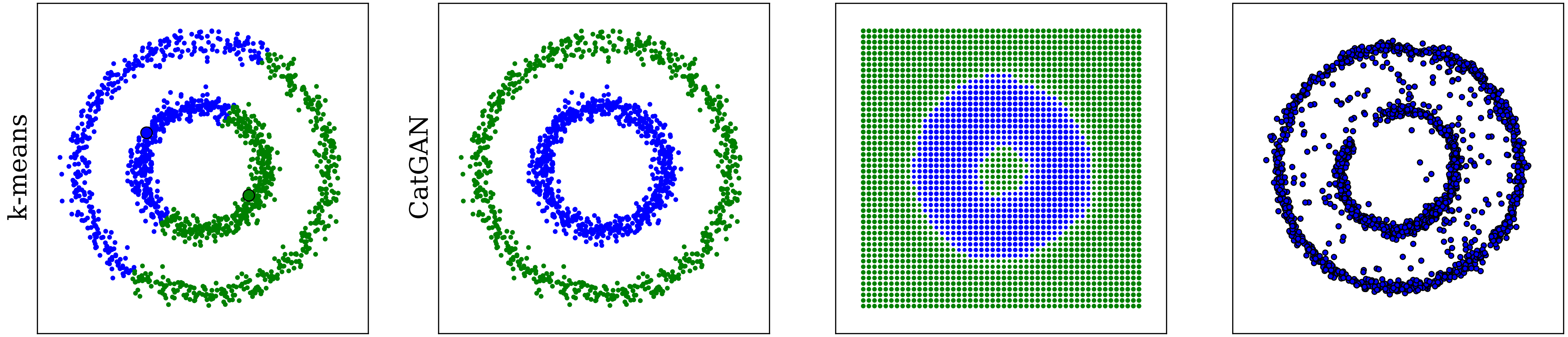
# Data imputation | In-painting | Denoising



# Semi-supervised Classification





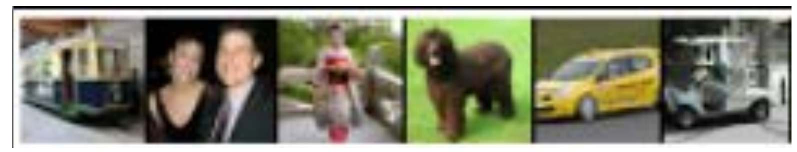

data + class assignment

decision boundaries

generated examples

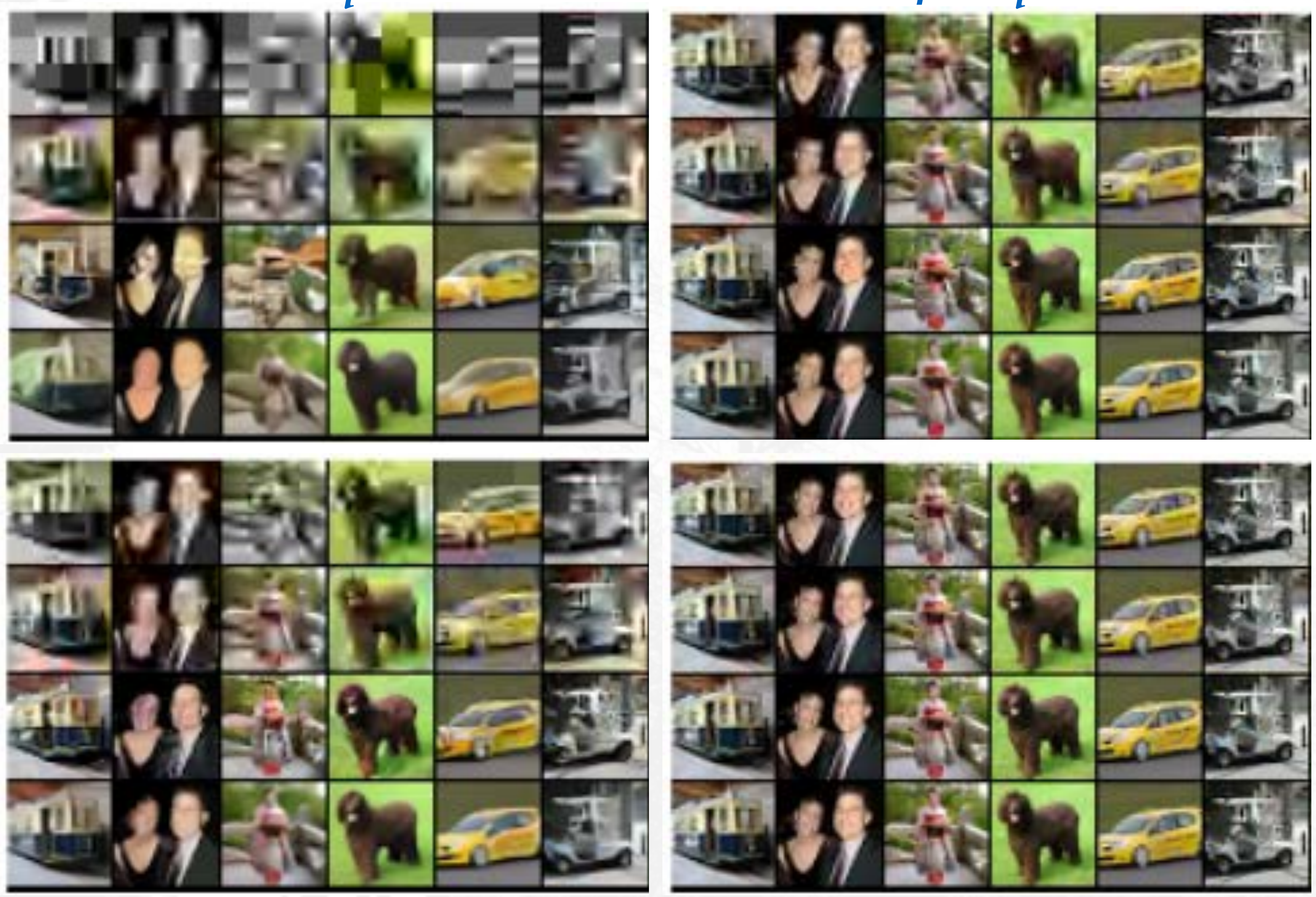
# Communication and Compression

Original Image



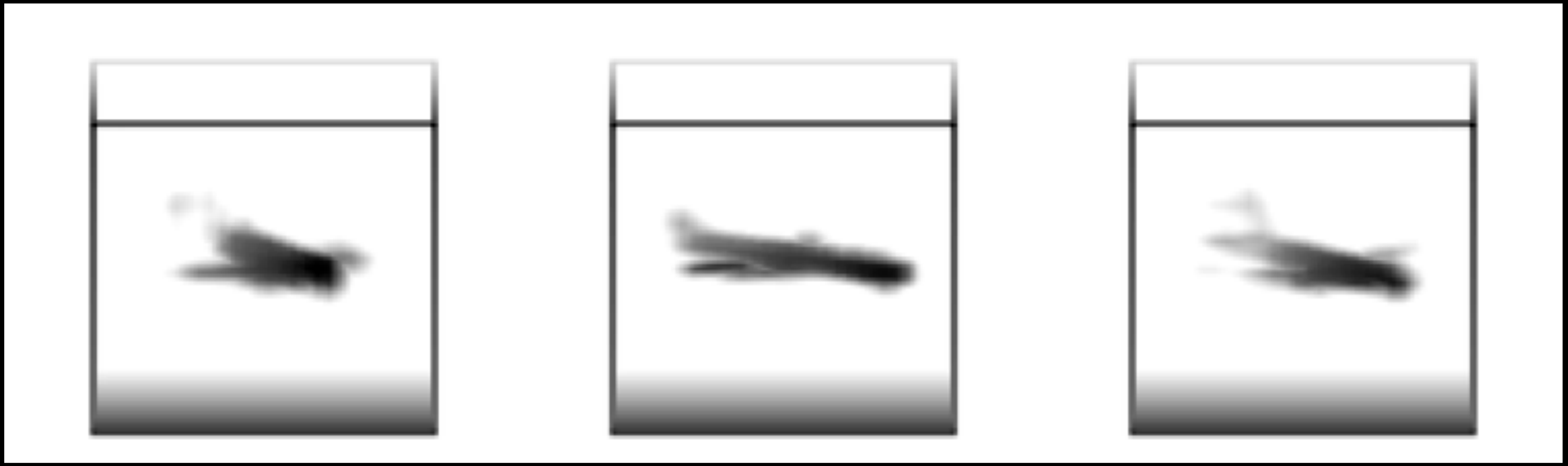
*0.1 bits/pixel* | *0.4 bits/pixel*

jpeg
jpeg 2000
generative
mean
jpeg
jpeg 2000
generative
mean

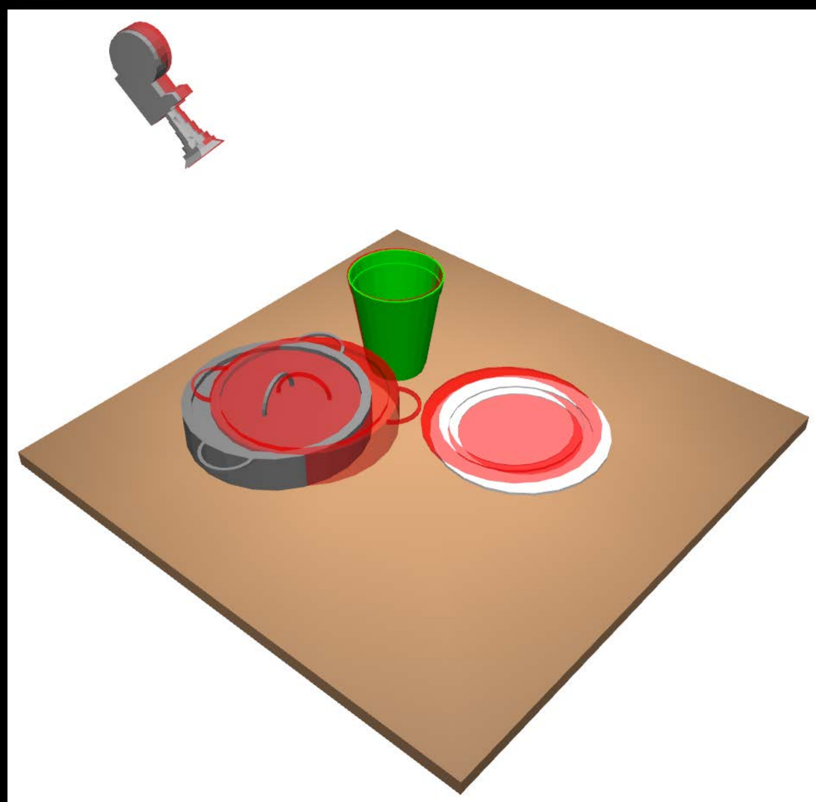


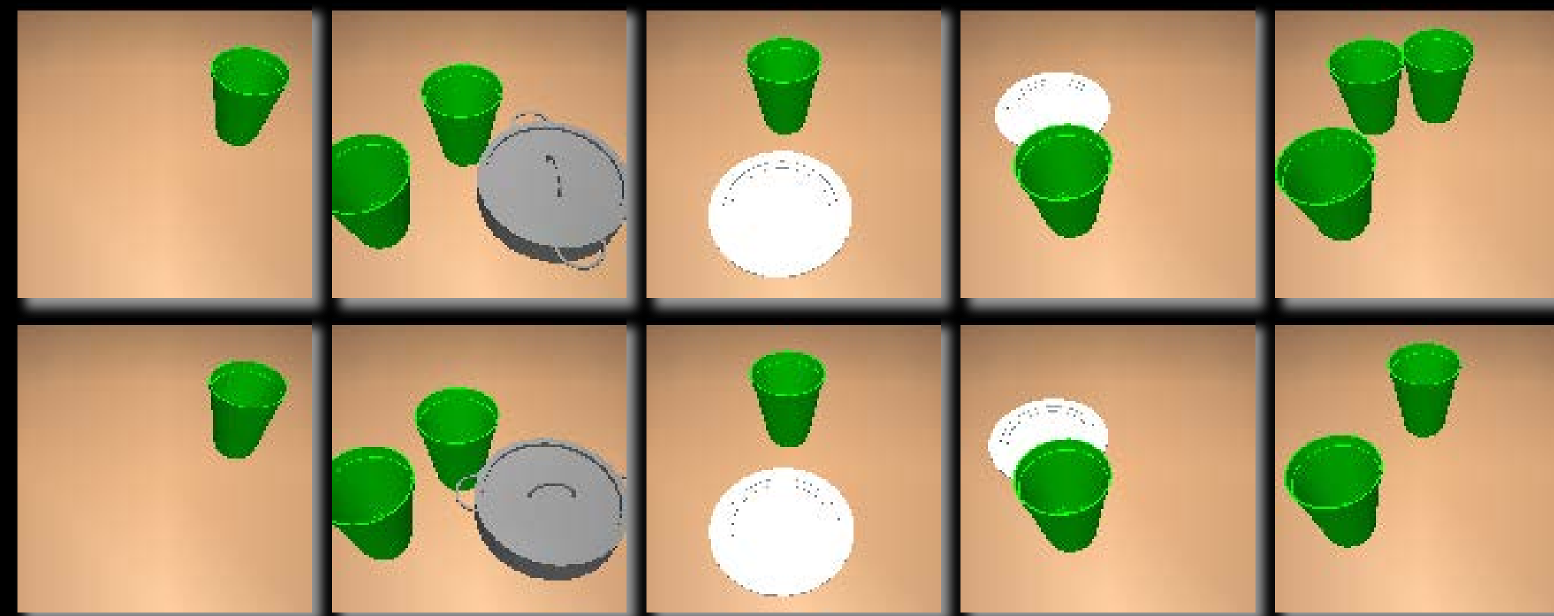
*0.2 bits/pixel* | *0.8 bits/pixel*

# 3D Scene Generation

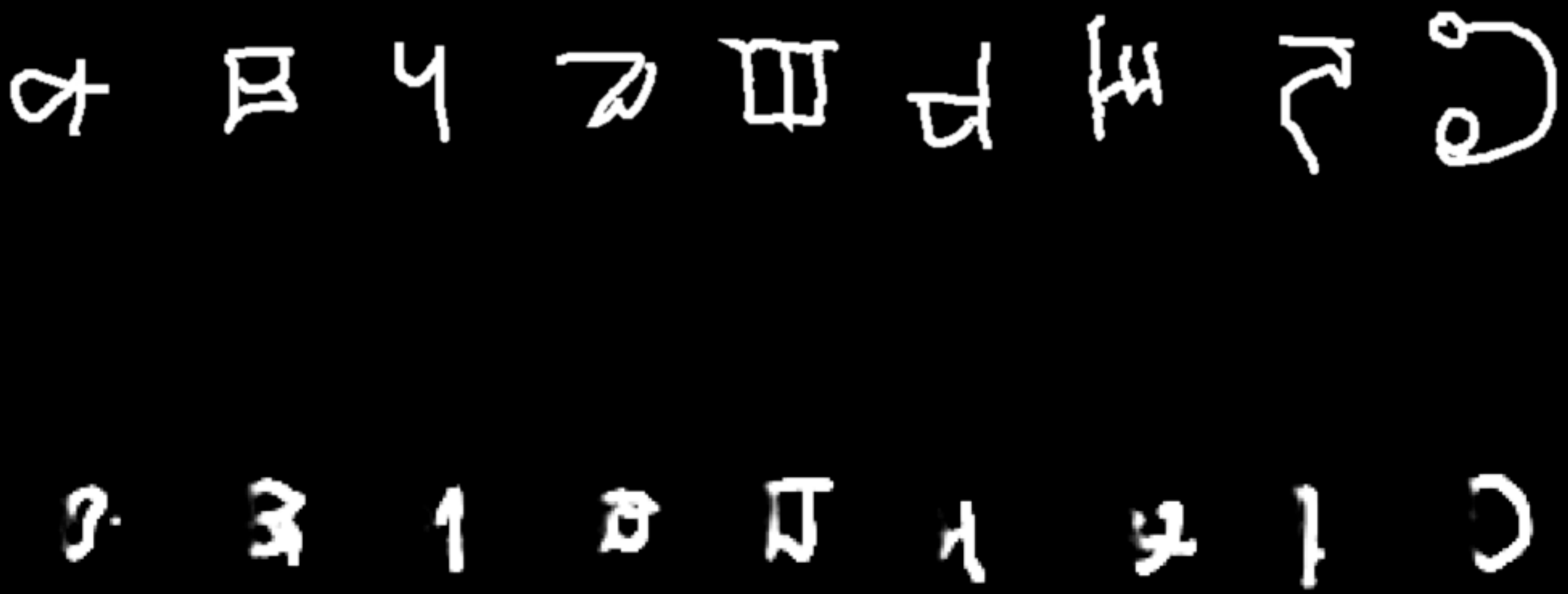


# Rapid Scene Understanding



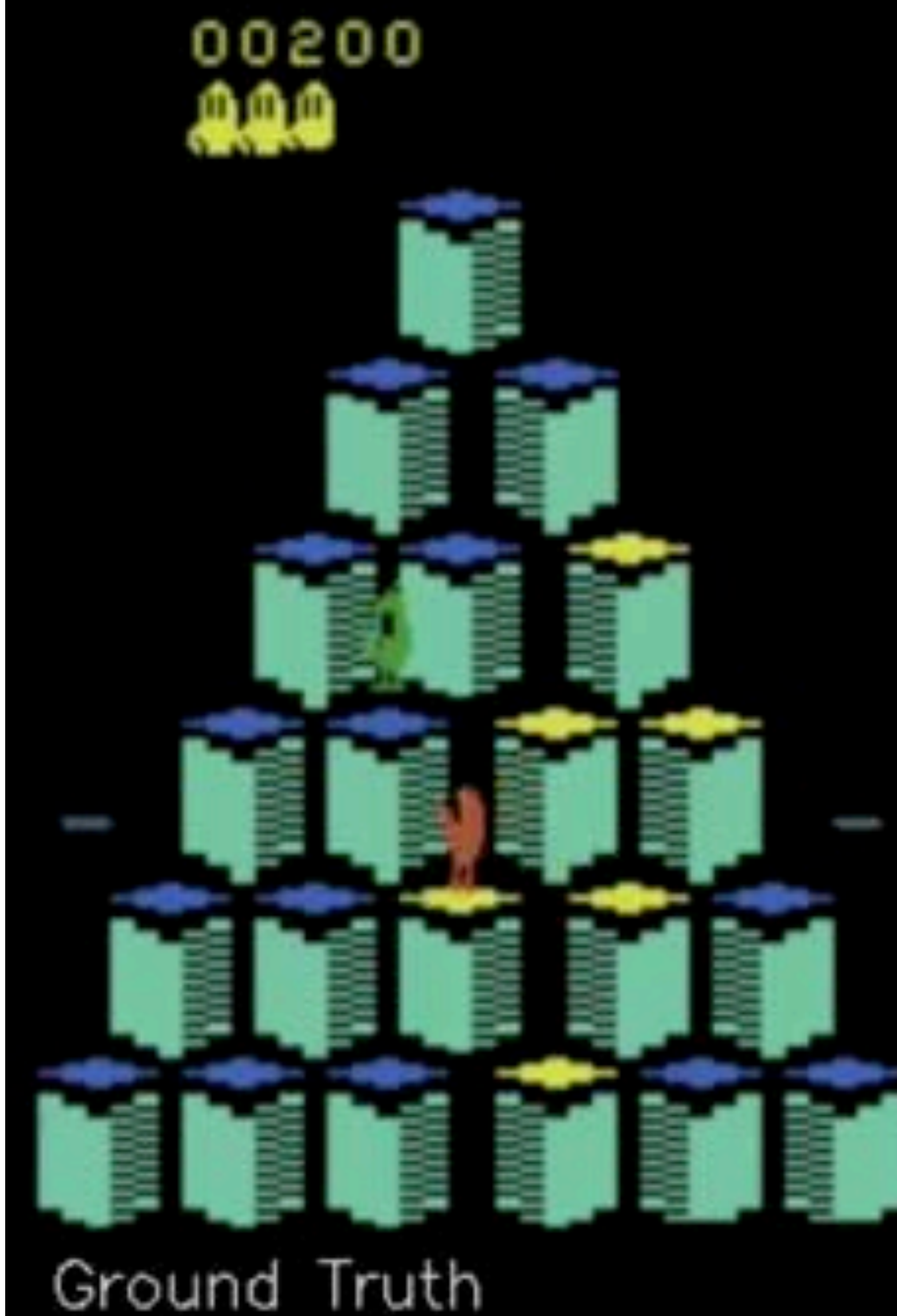


# One-shot Generalisation



# Environment Simulation

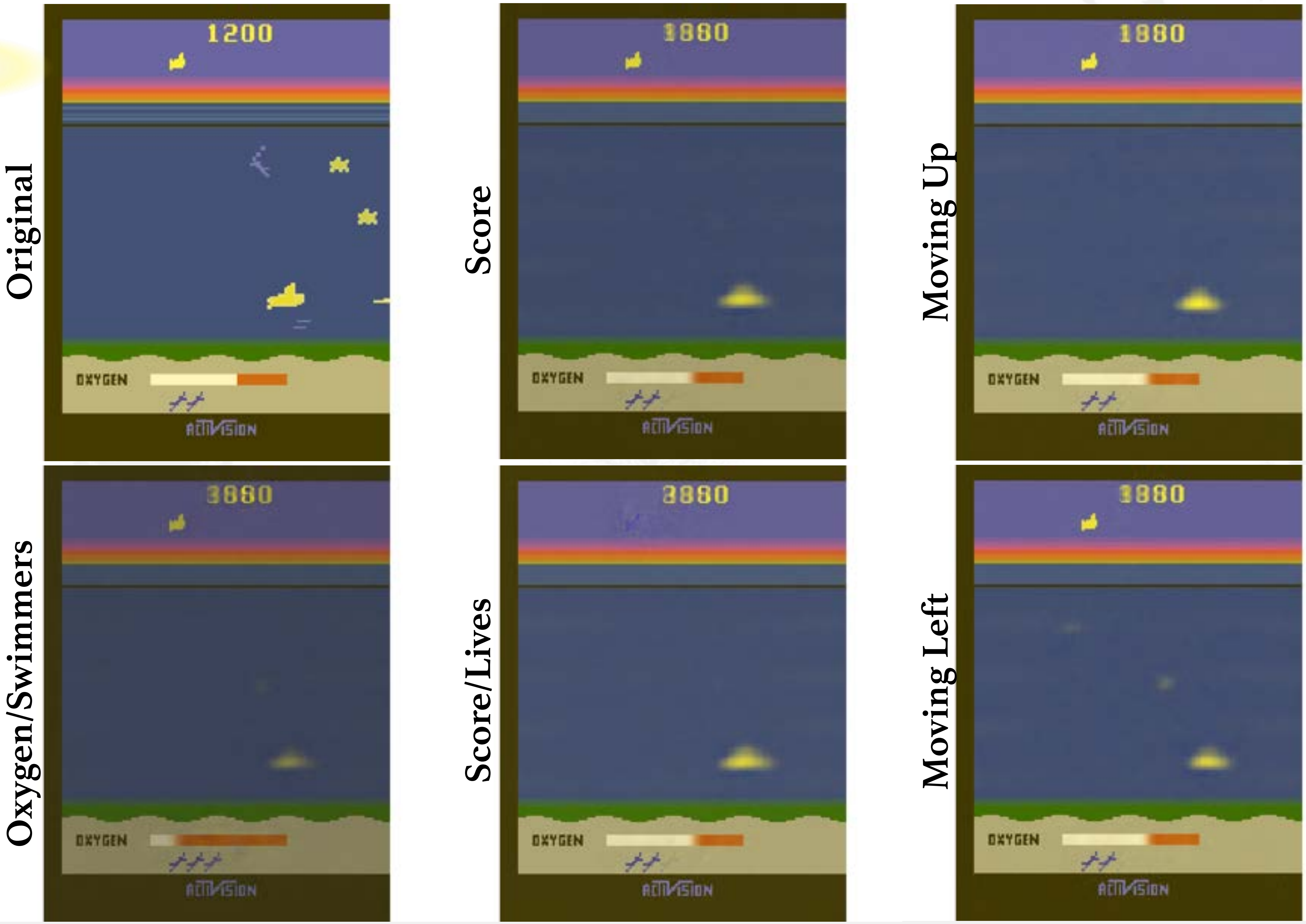

*Action-dependent simulator*



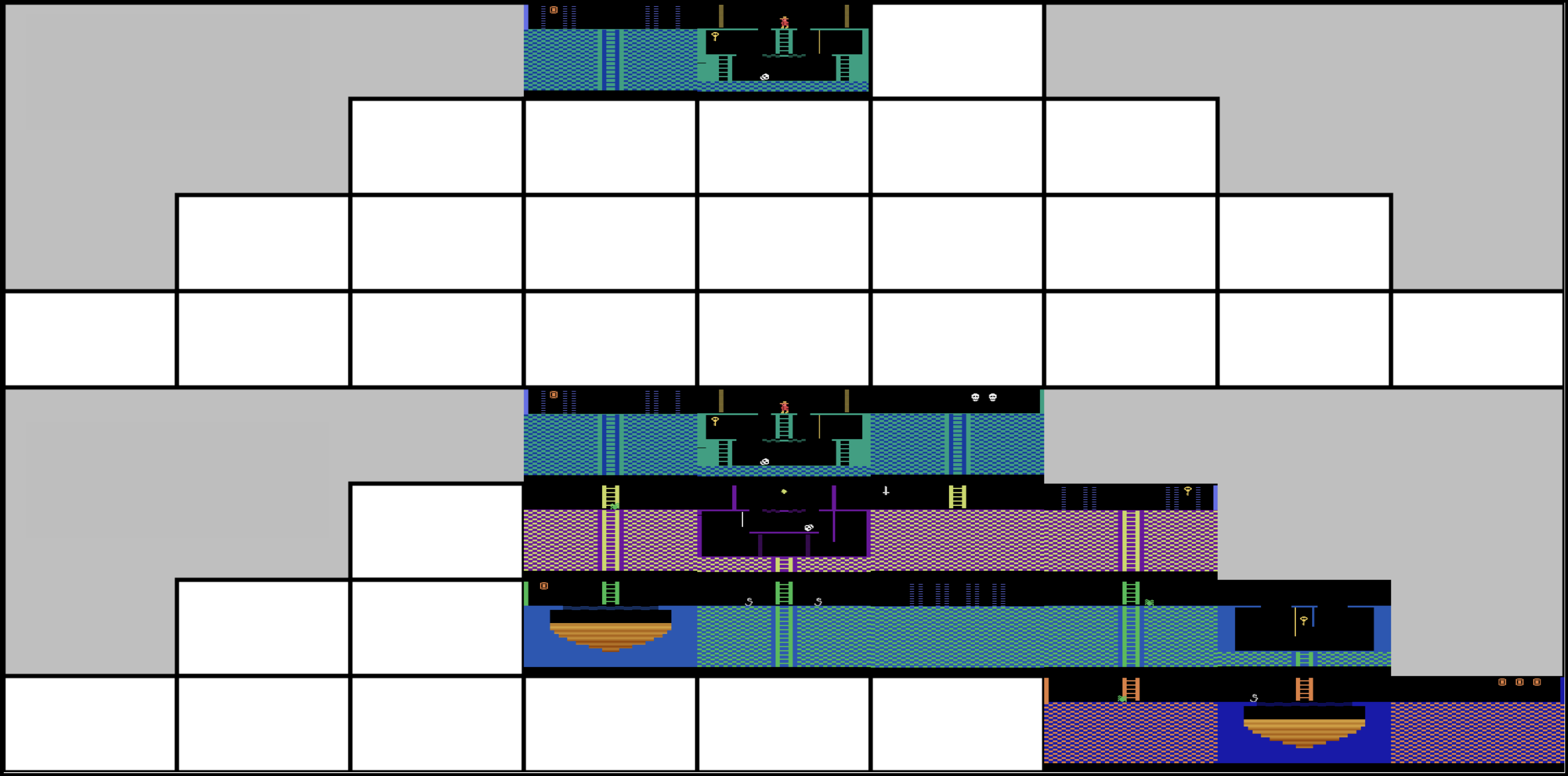

*Truth from Emulator*

# Representation Learning for Control

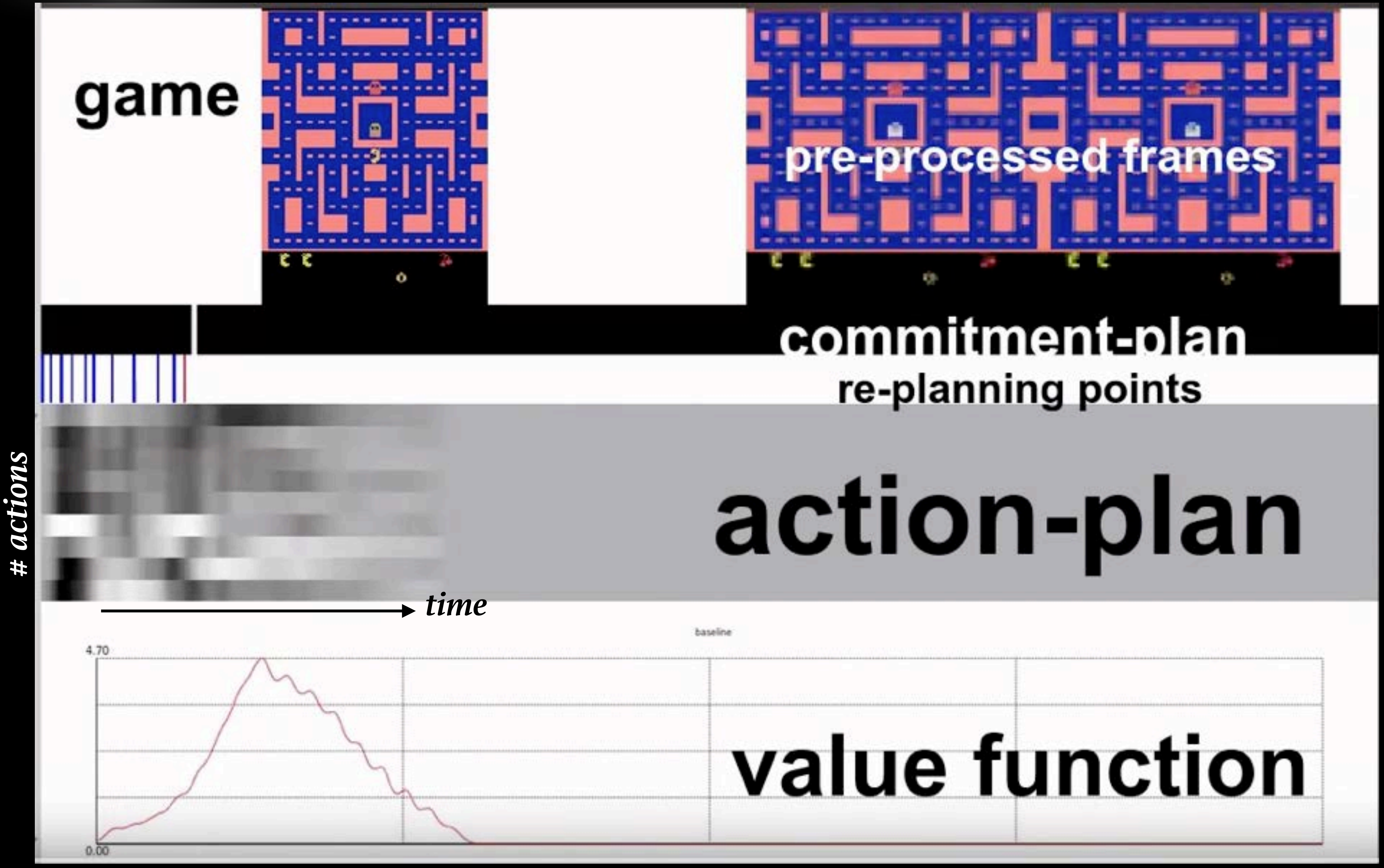
# Visual Concept Learning



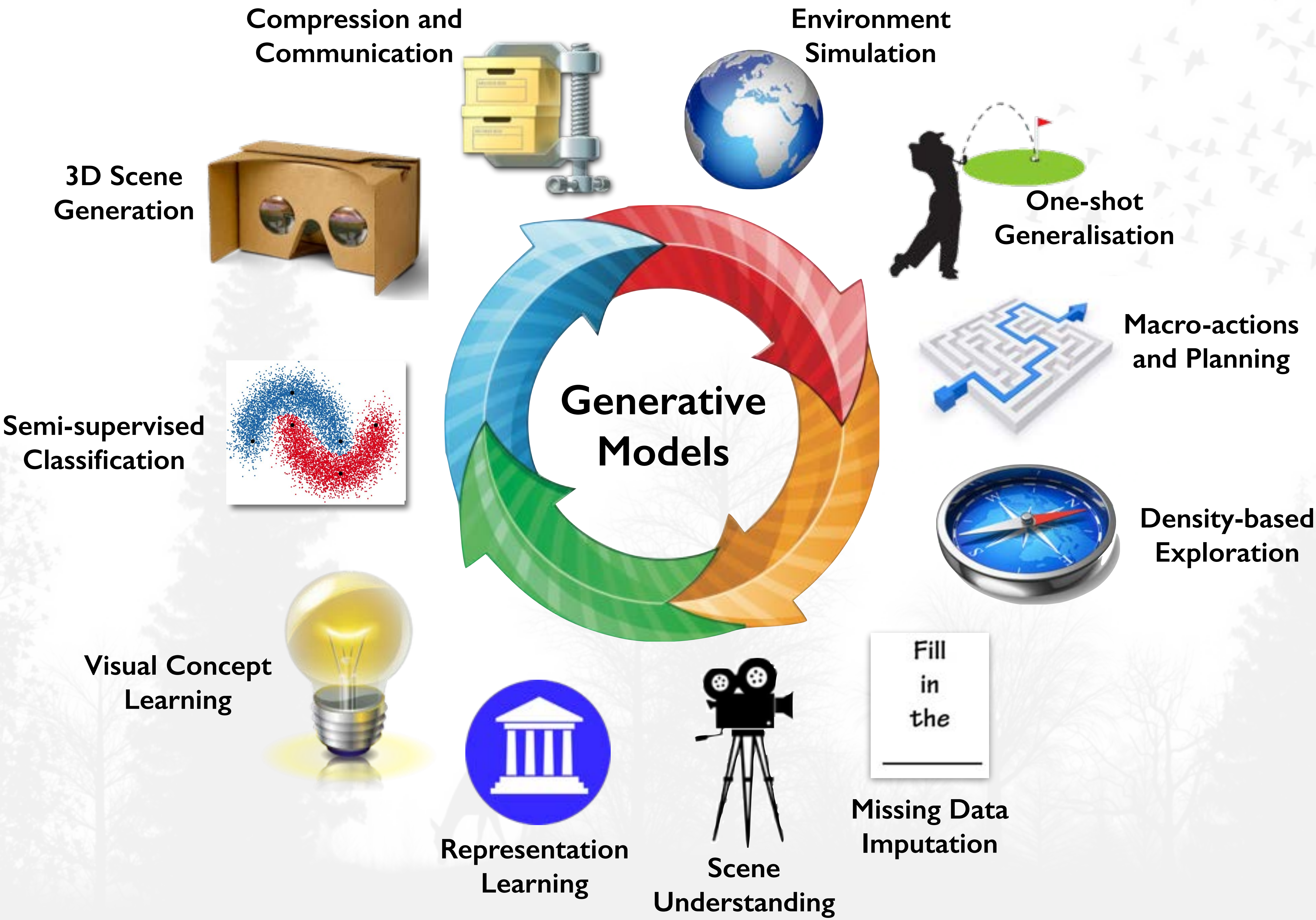
# Density-based Exploration



# Macro-actions and Planning

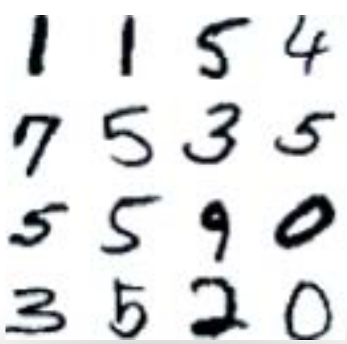


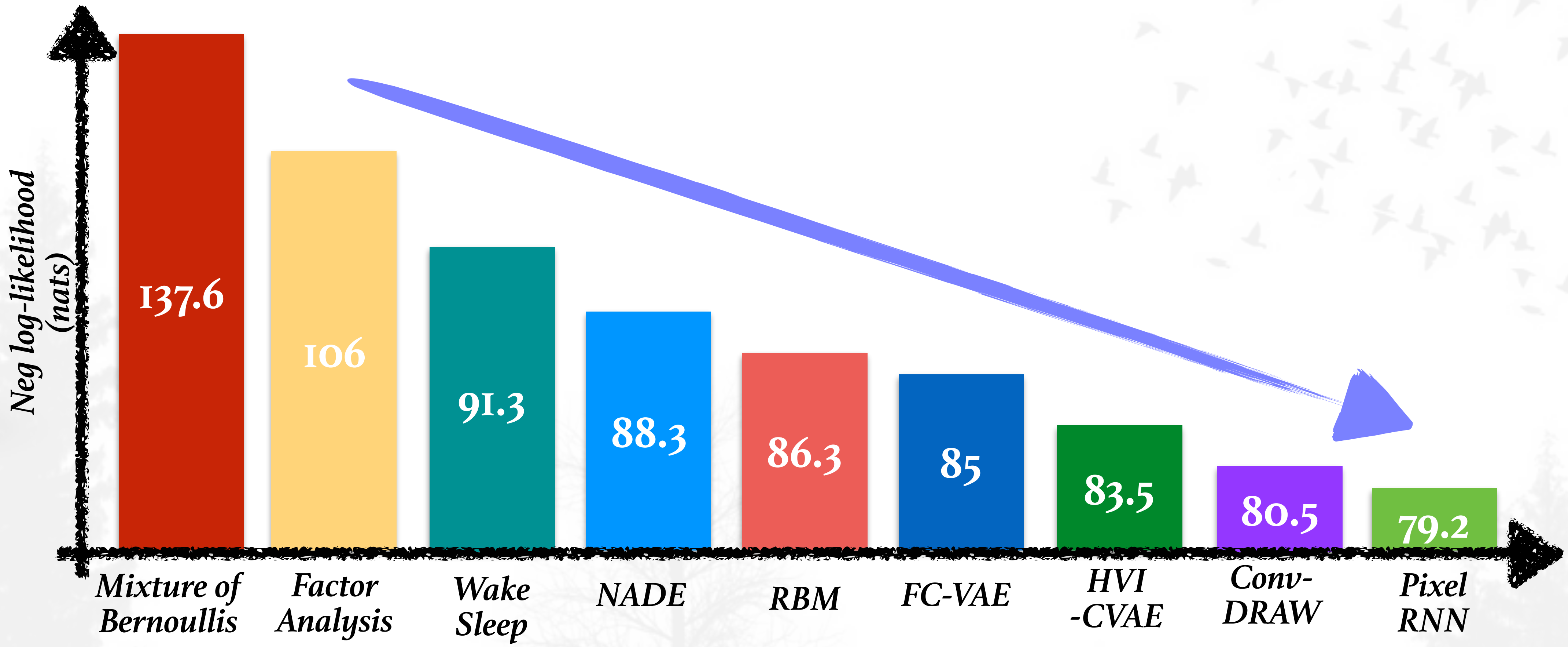
# Successful Applications of Generative Models



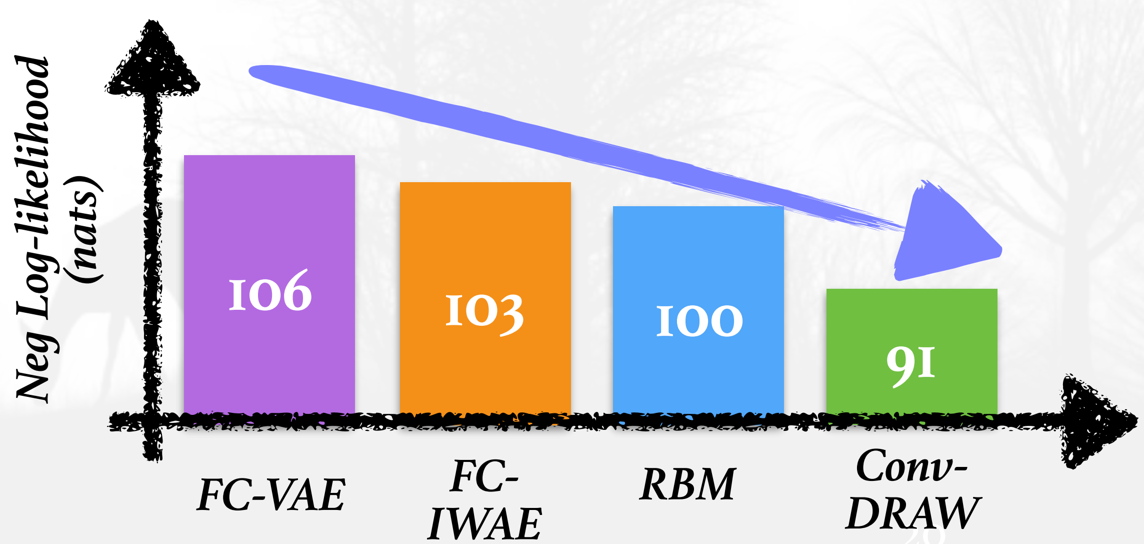
# Progress in Generative Models

## MNIST

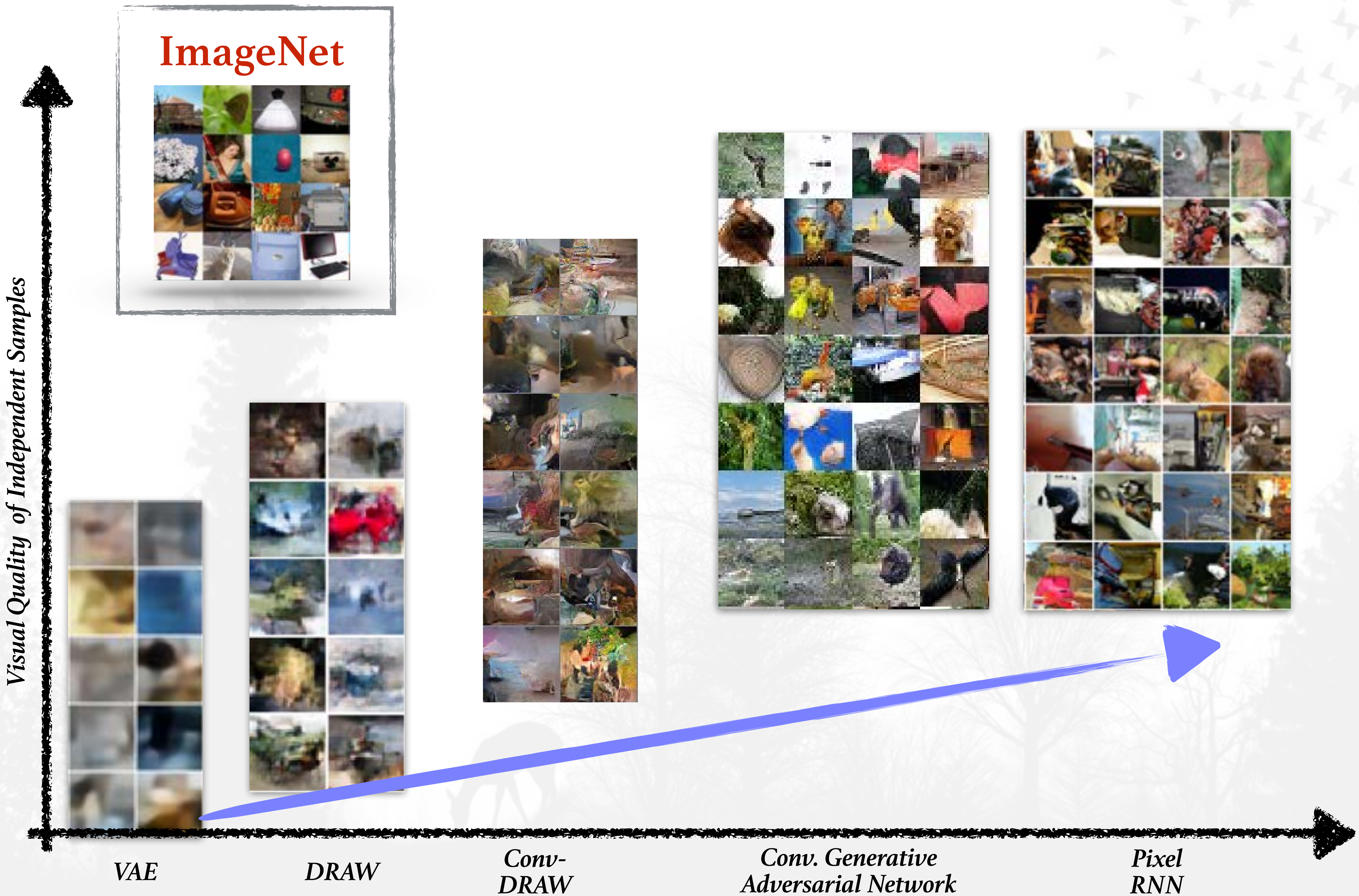




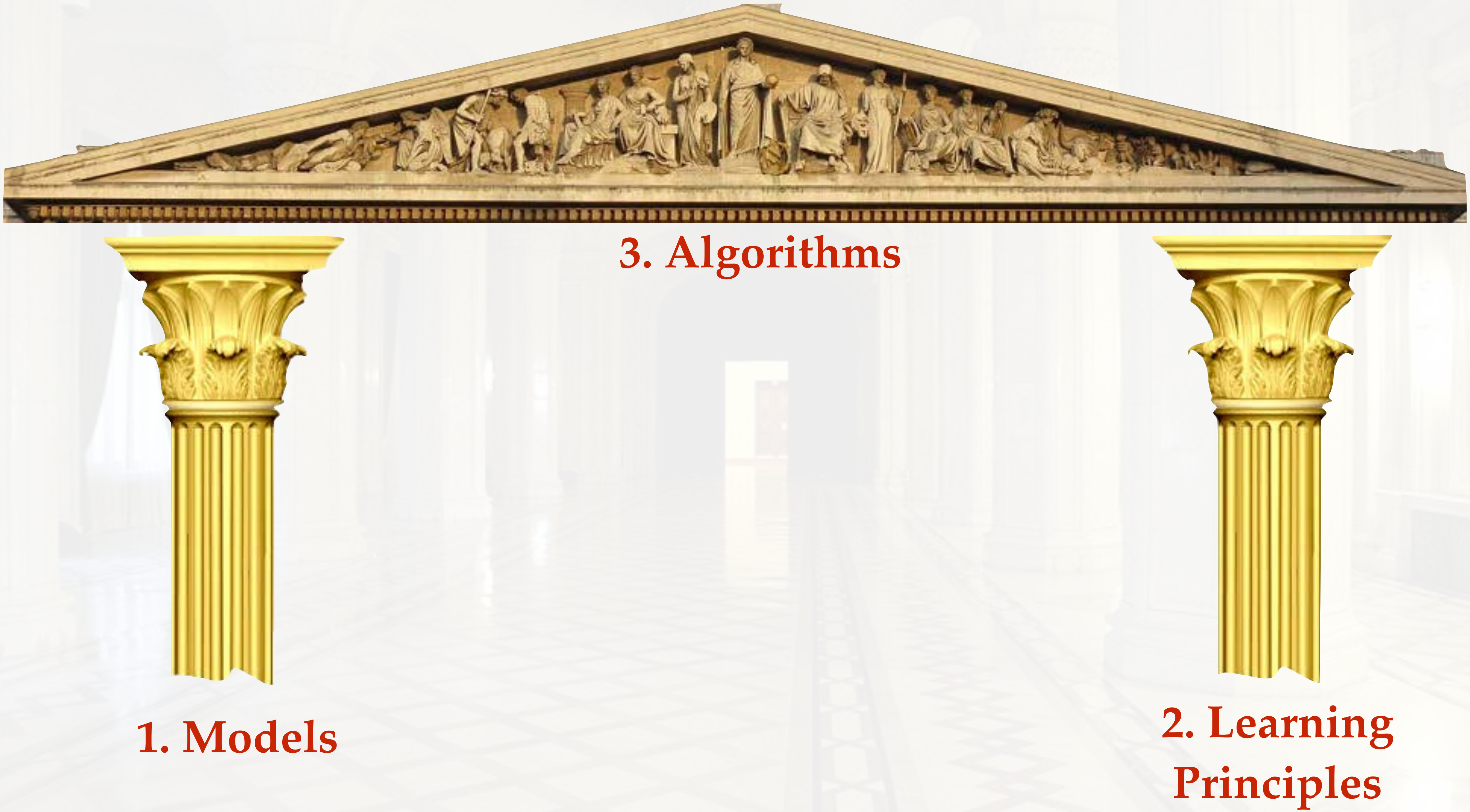

## Omniglot



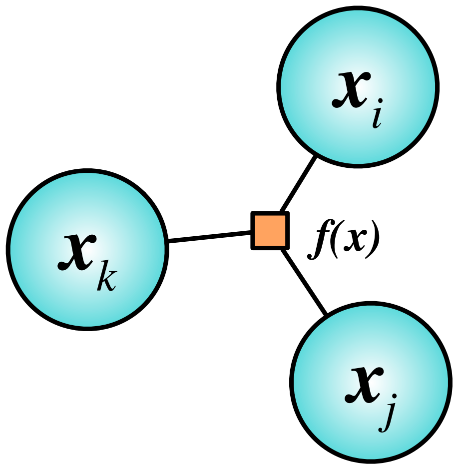
# Progress in Generative Models



# Machine Learning Framework

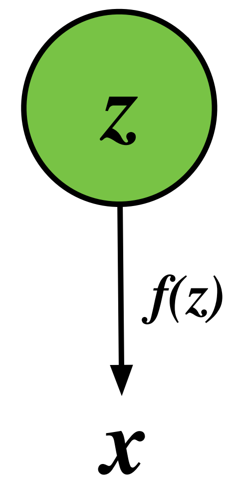


# Types of Generative Models




## Fully-observed models

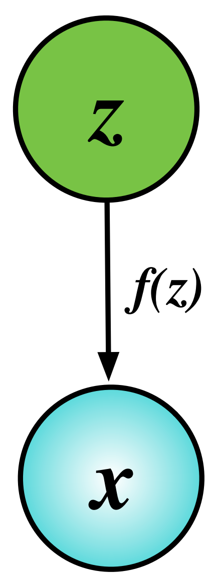
*Model observed data directly without introducing any new unobserved local variables.*




## Transformation models

*Model data as a transformation of an unobserved noise source using a parameterised function.*

**Models**




## Latent variable models

*Introduce an unobserved random variable for every observed data point to explain hidden causes.*

# Smorgasbord of Learning Principles

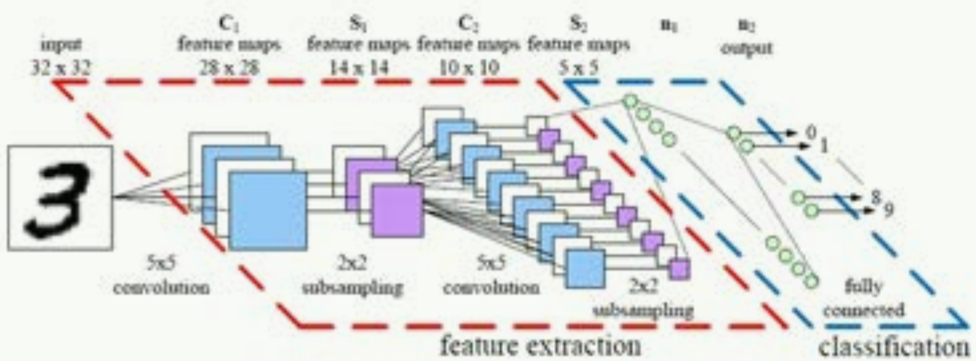
Learning Principles

*For a given model, there are*
*many competing inference methods.*

- Exact methods (conjugacy, enumeration)
- Numerical integration (Quadrature)
- Generalised method of moments
- **Maximum likelihood (ML)**
- **Maximum a posteriori (MAP)**
- Laplace approximation
- Integrated nested Laplace approximations (INLA)
- **Expectation Maximisation (EM)**
- Monte Carlo methods (MCMC, SMC, ABC)
- Noise contrastive estimation (NCE)
- Cavity Methods (EP)
- **Variational methods**

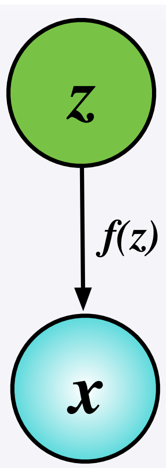
# Combining Models and Inference

*A given model and learning principle can be implemented in many ways.*



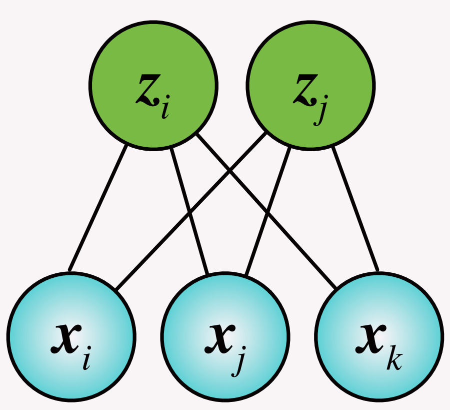

*Convolutional neural network*
*+ penalised maximum likelihood*

- Optimisation methods (SGD, Adagrad)
- Regularisation (L1, L2, batchnorm, dropout)




*Latent variable model*
*+ variational inference*

- VEM algorithm
- Expectation propagation
- Approximate message passing
- *Variational auto-encoders*




*Restricted Boltzmann Machine*
*+ maximum likelihood*

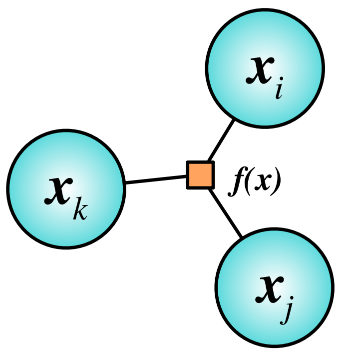
- Contrastive Divergence
- Persistent Contrastive Divergence
- Parallel Tempering
- Natural gradients

# Part II

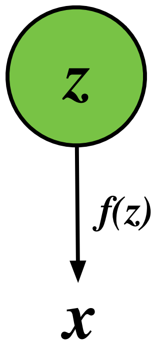
# A Model for Every Occasion

Explore three classes of generative models, their inductive biases, and implications for learning and algorithm design.

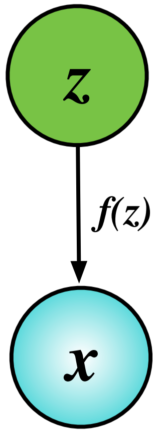
# Types of Generative Models




**Fully-observed models**




**Transformation models**




**Latent variable models**

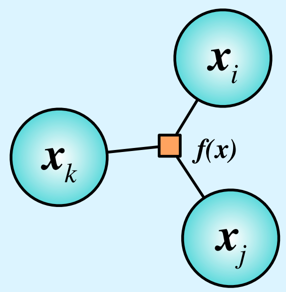
### Design Dimensions

- ***Data:*** binary, real-valued, nominal, strings, images.
- ***Dependency***: independent, sequential, temporal, spatial.
- ***Representation***: continuous or discrete
- ***Dimension***: parametric or non-parametric

---

- Computational complexity
- Modelling capacity
- Bias, uncertainty, calibration
- Interpretability

# Fully-observed Models



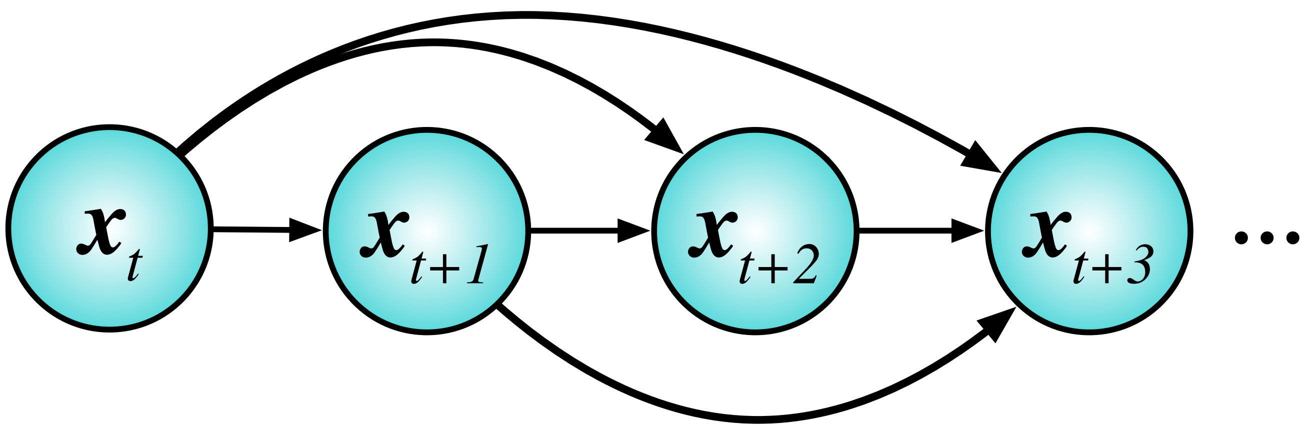

## Fully-observed models

*Model observed data directly **without** introducing any new unobserved **local variables**.*

*Model Parameters are **global variables.***

*Stochastic activations & unobserved random variables are **local variables.***

Markov Models



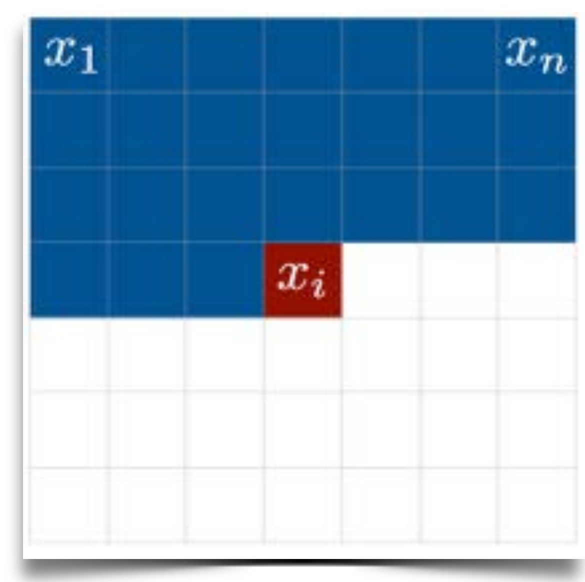

$$x_1 \sim \mathrm{Cat}(x_1|\pi)$$

$$x_2 \sim \mathrm{Cat}(x_2|\pi(\mathbf{x}_1))$$

$$\cdots$$

$$x_i \sim \mathrm{Cat}(x_i|\pi(\mathbf{x}_{<n}))$$

$$p(\mathbf{x}) = \prod_i p(x_i|f(\mathbf{x}_{<i};\boldsymbol{\theta}))$$




All conditional probabilities described by deep networks.

# Fully-observed Models

## Properties

- \+ Can directly encode how observed points are related.
- \+ *Any data* type can be used
- \+ For directed graphical models:
  - \+ **Parameter learning simple:** Log-likelihood is directly computable, no approximation needed.
  - \+ Easy to scale-up to large models, many optimisation tools available.
  - \- Order sensitive.
- \- For undirected models,
  - \- **Parameter learning difficult:** Need to compute normalising constants.
- \- **Generation can be slow:** iterate through elements sequentially, or using a Markov chain.

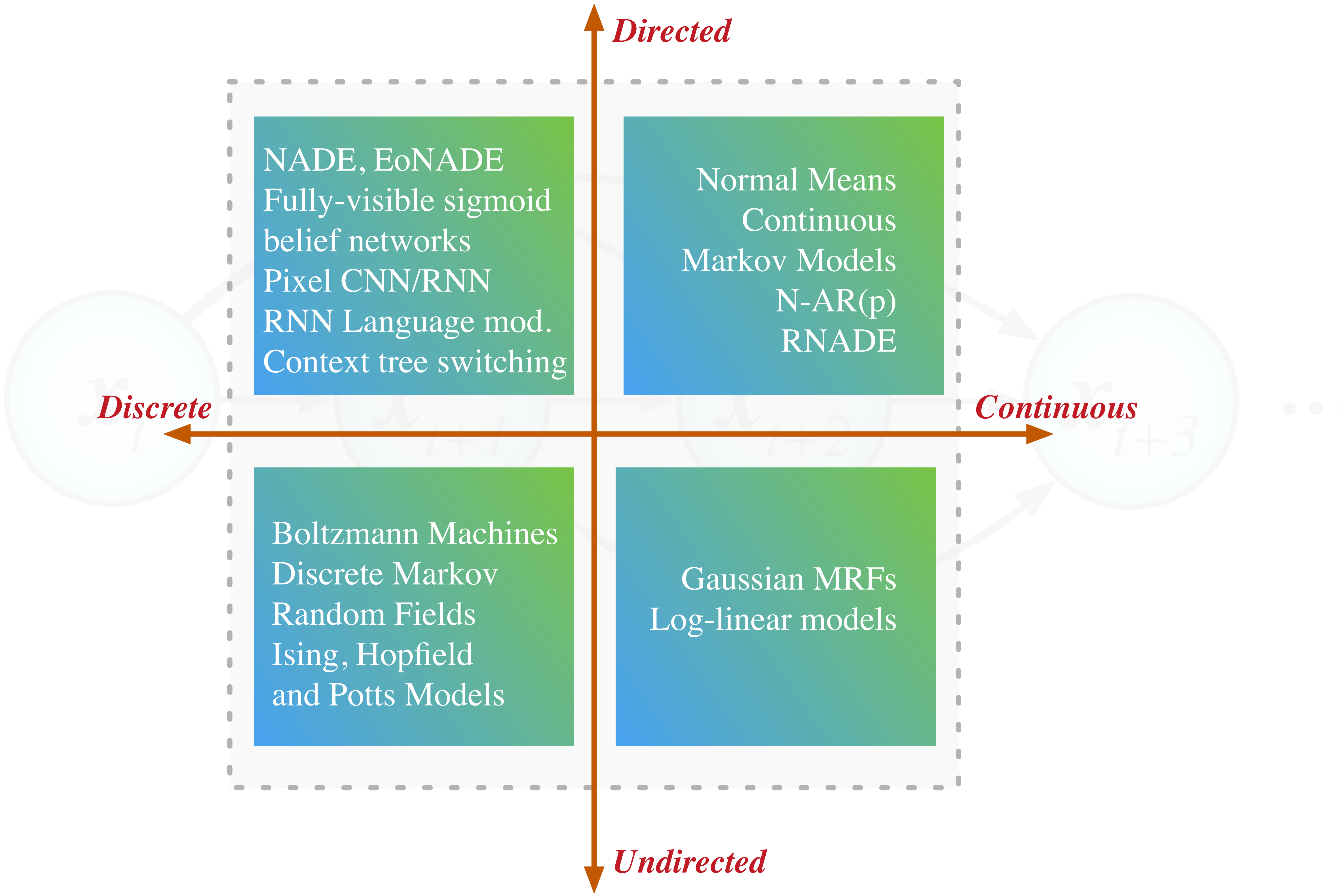
*Pixel CNN*

White Whale

Hartebeest

# Model-space Visualisation

***Fully-observed models***

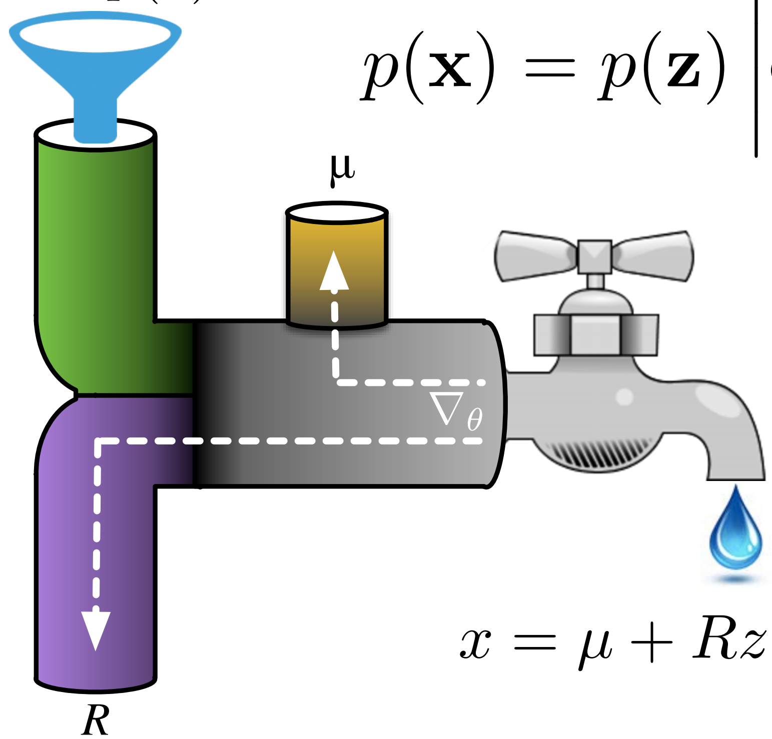


# Transformation Models

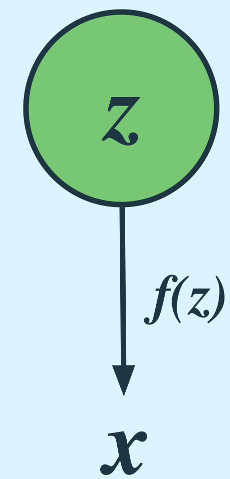
*Change of variables for invertible functions*

$z \sim p(z)$

$$p(\mathbf{x}) = p(\mathbf{z}) \left| \det \frac{\partial f}{\partial \mathbf{z}} \right|^{-1}$$




$x = \mu + Rz$



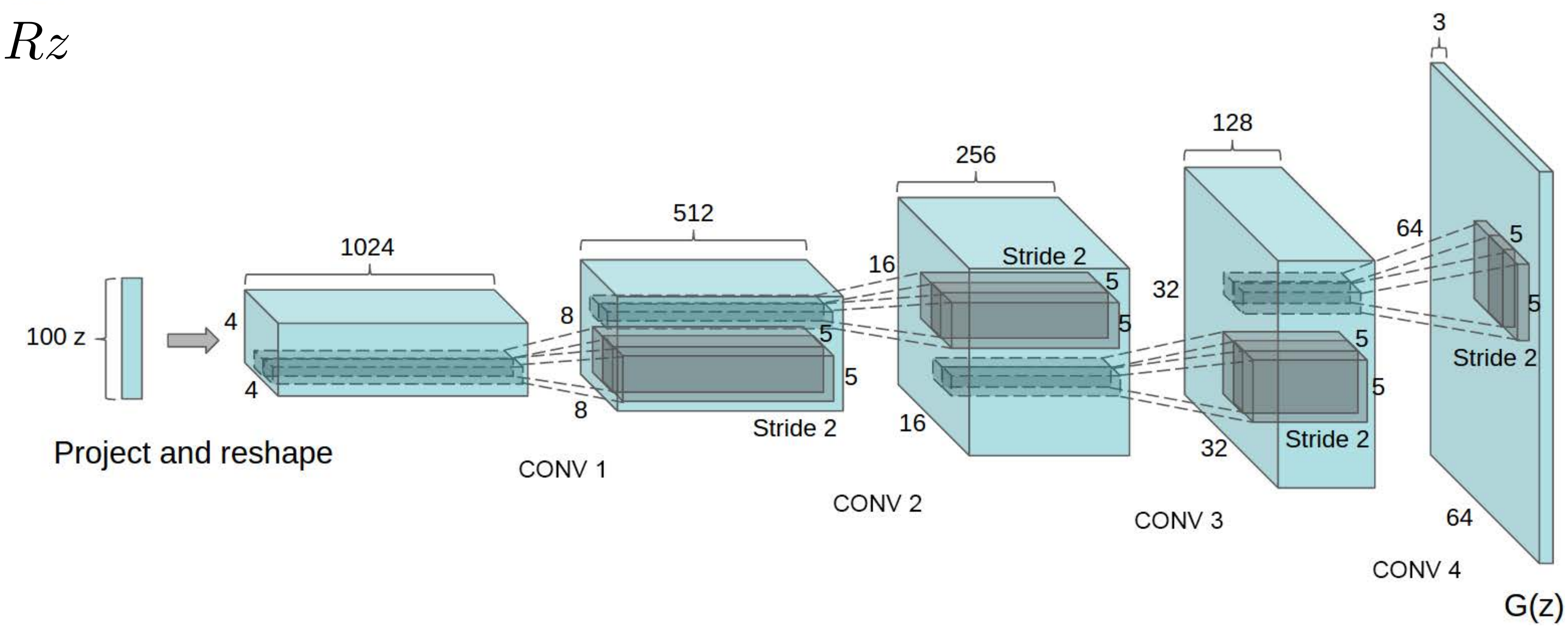

## Transformation models

*Transform an unobserved* ***noise source*** *using a parameterised* ***function****.*

## *Generator Networks*

$$\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

$$\mathbf{x} = f(\mathbf{z}; \boldsymbol{\theta})$$




*The transformation function is parameterised by a linear or deep network (fully-connected, convolutional or recurrent).*

# Transformation Models

## Properties

+ Easy sampling
+ Easy to compute expectations without knowing final distribution.
+ Can exploit with large-scale classifiers and convolutional networks.
- ***Difficult to satisfy constraints***: Difficult to maintain invertibility, and challenging optimisation.
- ***Lack of noise model*** (likelihood):
    - Difficult to extend to generic data types
    - Difficult to account for noise in observed data.
    - Hard to compute marginalised likelihood for model scoring, comparison and selection.

*Convolutional generative adversarial network*

# Model-space Visualisation

***Transformation models***

Stochastic
Differential Equations
Hamiltonian and
Langevin SDE
Diffusion Models
Non- and volume
preserving flows

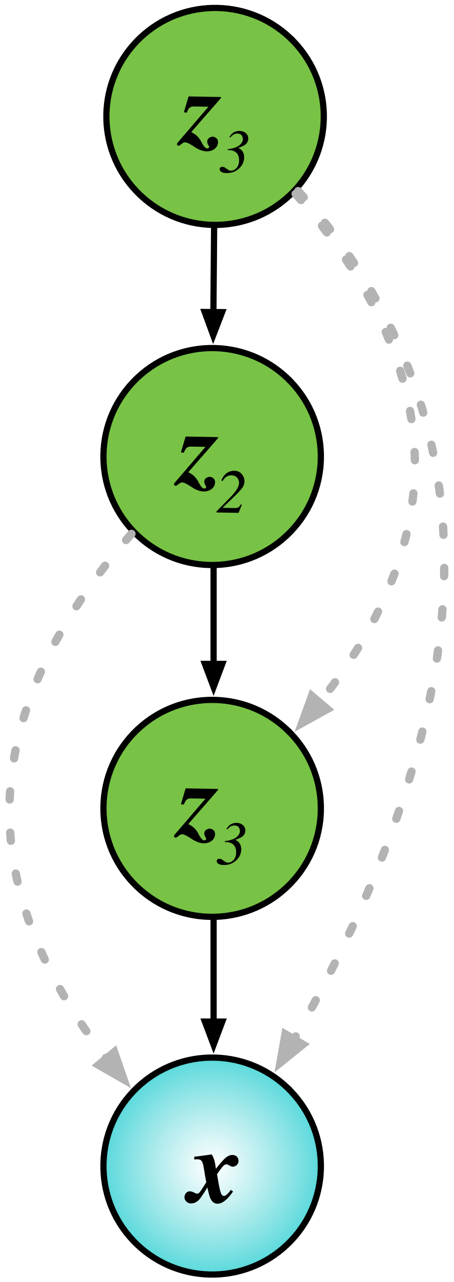
One-liners and
inverse sampling
Distrib. warping
Normalising flows
GAN generator nets
Non- and volume
preserving transforms

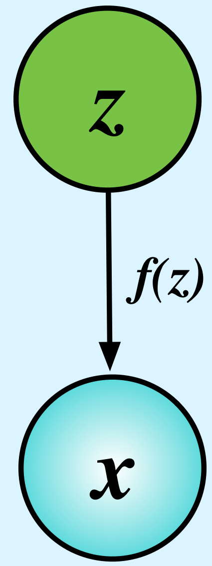
***Diffusions***
***Continuous time***

***Functions***
***Discrete time***

# Latent Variable Models

Deep Latent Gaussian Model




**Latent variable models**

*Introduce an unobserved* ***local random variables*** *that represents hidden causes.*

$$\mathbf{z}_3 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

$$\mathbf{z}_2 | \mathbf{z}_3 \sim \mathcal{N}(\mu(\mathbf{z}_3), \Sigma(\mathbf{z}_3))$$

$$\mathbf{z}_1 | \mathbf{z}_2 \sim \mathcal{N}(\mu(\mathbf{z}_2), \Sigma(\mathbf{z}_2))$$

$$\mathbf{x} | \mathbf{z}_1 \sim \mathcal{N}(\mu(\mathbf{z}_1), \Sigma(\mathbf{z}_1))$$

# Latent Variable Models

**Properties**

- \+ Easy sampling.
- \+ Easy way to include hierarchy and depth.
- \+ Easy to encode structure believed to generate the data
- \+ Avoids order dependency assumptions: marginalisation of latent variables induces dependencies.
- \+ Latents provide compression and representation the data.
- \+ Scoring, model comparison and selection possible using the marginalised likelihood.
- \- Inversion process to determine latents corresponding to a input is difficult in general
- \- Difficult to compute marginalised likelihood requiring approximations.
- \- Not easy to specify rich approximations for latent posterior distribution.

*Convolutional DRAW*

# Model-space Visualisation

## *Latent variable models*

Cascaded Indian Buffet process
Hierarchical Dirichlet process

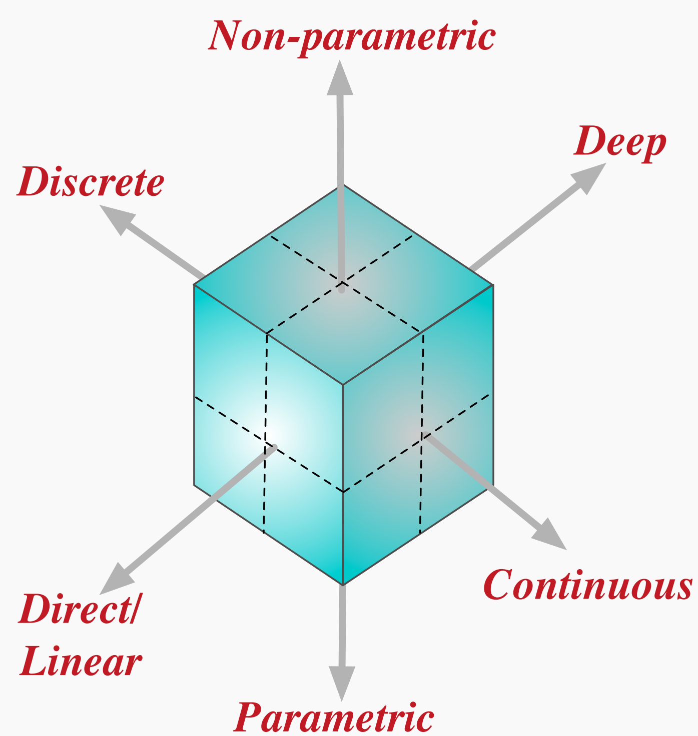
Deep Nonparametric Discrete

Sigmoid Belief Net
Deep auto-regressive networks (DARN)

Deep Parametric Discrete

Indian buffet process
Dirichlet process mixture

Direct Nonparametric Discrete




Deep Gaussian processes
Recurrent Gaussian Process
GP State space model

Deep Nonparametic Continuous

Hidden Markov Model
Discrete LVM
Sparse LVMs

Linear Parametric Discrete

Nonlinear factor analysis
Nonlinear Gaussian belief network
Deep Latent Gaussian (VAE, DRAW)

Deep Parametric Continuous

PCA, factor analysis
Independent components analysis
Gaussian LDS
Latent Gauss Field

Linear Parametric Continuous

Gaussian process LVM

Direct Nonparametric Continuous

# Part III



# Inference and Learning

Principles and approximations that can be used to drive learning in different types of models.

- Model evidence
- Two-sample testing

# Inferential Problems

***Common inference problems are:***

**Evidence Estimation**

$$p(\mathbf{x}) = \int p(\mathbf{x}, \mathbf{z}) d\mathbf{z}$$

**Moment Computation**

$$\mathbb{E}[f(\mathbf{z})|\mathbf{x}] = \int f(\mathbf{z}) p(\mathbf{z}|\mathbf{x}) d\mathbf{z}$$

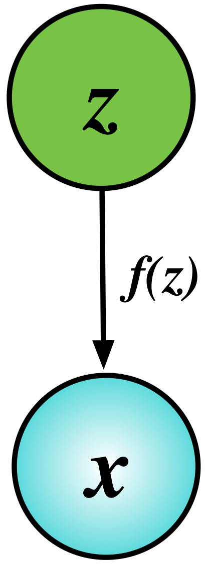
**Prediction**

$$p(\mathbf{x}_{t+1}) = \int p(\mathbf{x}_{t+1}|\mathbf{x}_t) p(\mathbf{x}_t) d\mathbf{x}_t$$

**Hypothesis Testing**

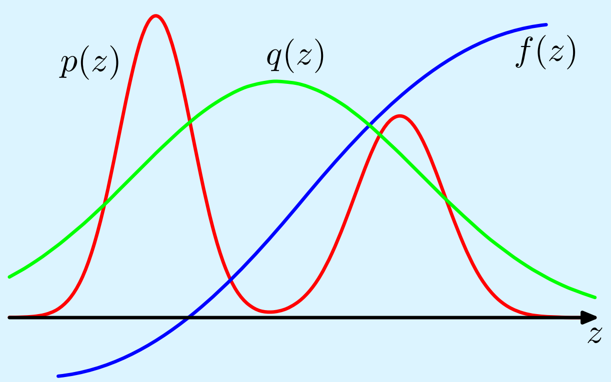
$$\mathcal{B} = \log p(\mathbf{x}|H_1) - \log p(\mathbf{x}|H_2)$$

# Bayesian Model Evidence




***Model evidence (or marginal likelihood, partition function)***: Integrating out any global and local variables enables model scoring, comparison, selection, moment estimation, normalisation, posterior computation and prediction.

***We take steps to improve the model evidence for given data samples.***



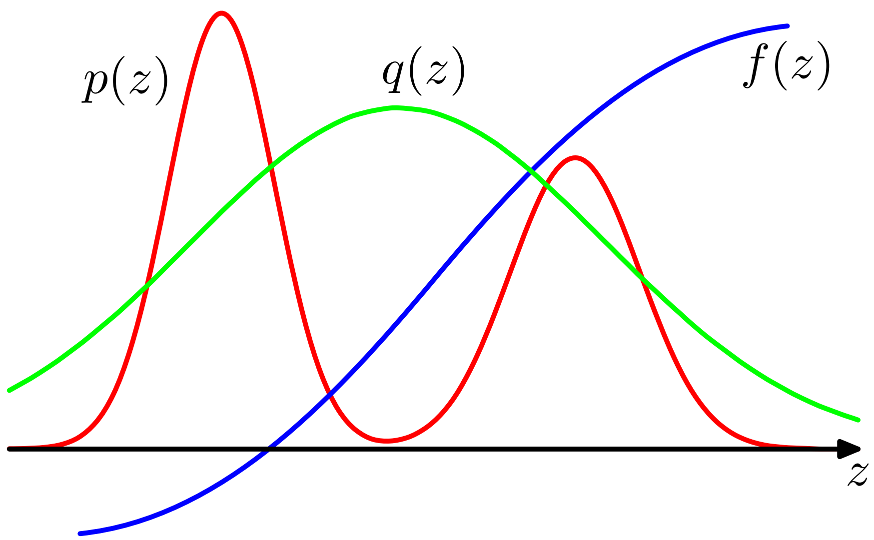

**Learning principle: Model Evidence**

$$p(\mathbf{x}) = \int p(\mathbf{x}, \mathbf{z}) d\mathbf{z}$$

Integral is intractable in general and requires approximation.

***Basic idea:*** Transform the integral into an expectation over a simple, known distribution.

# Importance Sampling




Integral problem
$$p(\mathbf{x}) = \int p(\mathbf{x}|\mathbf{z})p(\mathbf{z})d\mathbf{z}$$

Proposal
$$p(\mathbf{x}) = \int p(\mathbf{x}|\mathbf{z})p(\mathbf{z})\frac{q(\mathbf{z})}{q(\mathbf{z})}d\mathbf{z}$$

Importance Weight
$$p(\mathbf{x}) = \int p(\mathbf{x}|\mathbf{z})\frac{p(\mathbf{z})}{q(\mathbf{z})}q(\mathbf{z})d\mathbf{z}$$

$$w^{(s)} = \frac{p(z)}{q(z)} \quad z^{(s)} \sim q(z)$$

Monte Carlo
$$p(\mathbf{x}) = \frac{1}{S}\sum_s w^{(s)}p(\mathbf{x}|\mathbf{z}^{(s)})$$

***Notation***

Always think of $q(z|x)$ but often will write $q(z)$ for simplicity.

***Conditions***

- $q(z|x)>0$, when $f(z)p(z) \neq 0$.
- Easy to sample from $q(z)$.

# Importance Sampling to Variational Inference

Integral problem

$$p(\mathbf{x}) = \int p(\mathbf{x}|\mathbf{z})p(\mathbf{z})d\mathbf{z}$$

Proposal

$$p(\mathbf{x}) = \int p(\mathbf{x}|\mathbf{z})p(\mathbf{z})\frac{q(\mathbf{z})}{q(\mathbf{z})}d\mathbf{z}$$

Importance Weight

$$p(\mathbf{x}) = \int p(\mathbf{x}|\mathbf{z})\frac{p(\mathbf{z})}{q(\mathbf{z})}q(\mathbf{z})d\mathbf{z}$$

Jensen's inequality

$$\log \int p(x)g(x)dx \geq \int p(x) \log g(x)dx$$

$$\log p(\mathbf{x}) \geq \int q(\mathbf{z}) \log \left( p(\mathbf{x}|\mathbf{z})\frac{p(\mathbf{z})}{q(\mathbf{z})} \right) d\mathbf{z}$$

$$= \int q(\mathbf{z}) \log p(\mathbf{x}|\mathbf{z}) - \int q(\mathbf{z}) \log \frac{q(\mathbf{z})}{p(\mathbf{z})}$$

Variational lower bound

$$\boxed{\mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}|\mathbf{z})] - KL[q(\mathbf{z})\|p(\mathbf{z})]}$$

# Variational Free Energy


$$\mathcal{F}(\mathbf{x}, q) = \underbrace{\mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}|\mathbf{z})]}_{\text{Reconstruction}} - \underbrace{KL[q(\mathbf{z})\|p(\mathbf{z})]}_{\text{Penalty}}$$

Approx. Posterior

Interpreting the bound:

- **Approximate posterior distribution *q(z|x)*:** Best match to true posterior *p(z|x)*, one of the unknown inferential quantities of interest to us.
- **Reconstruction cost**: The expected log-likelihood measures how well samples from *q(z|x)* are able to explain the data *x*.
- **Penalty:** Ensures that the explanation of the data *q(z|x)* doesn't deviate too far from your beliefs *p(z)*. A mechanism for realising Ockham's razor.

# Other Families of Variational Bounds

*Variational Free Energy*

$$\mathcal{F}(\mathbf{x}, q) = \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}|\mathbf{z})] - KL[q(\mathbf{z})\|p(\mathbf{z})]$$

*Multi-sample Variational Objective*

$$\mathcal{F}(\mathbf{x}, q) = \mathbb{E}_{q(z)}\left[\log \frac{1}{S}\sum_s \frac{p(\mathbf{z})}{q(\mathbf{z})} p(\mathbf{x}|\mathbf{z})\right]$$

*Renyi Variational Objective*

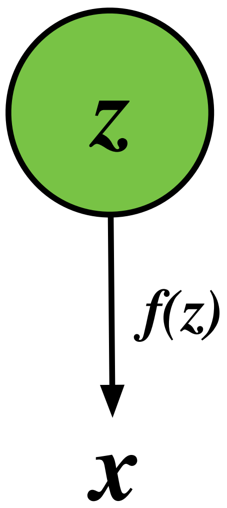
$$\mathcal{F}(\mathbf{x}, q) = \frac{1}{1-\alpha}\mathbb{E}_{q(z)}\left[\left(\log \frac{1}{S}\sum_s \frac{p(\mathbf{z})}{q(\mathbf{z})} p(\mathbf{x}|\mathbf{z})\right)^{1-\alpha}\right]$$

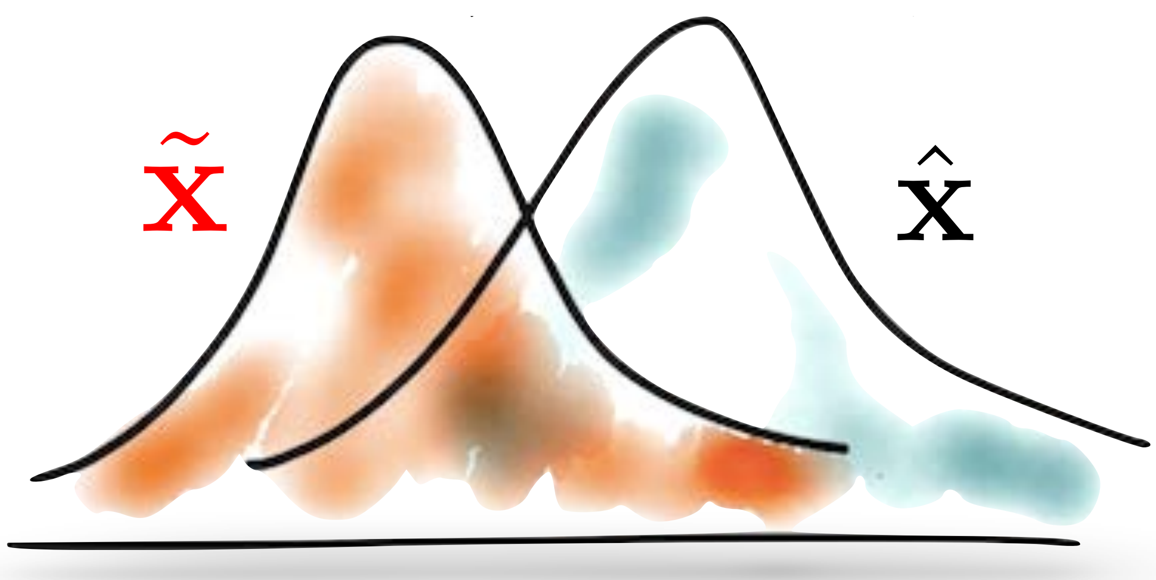
***Other generalised families exist. Optimal solution is the same for all objectives.***

# Bayesian Two-sample Testing

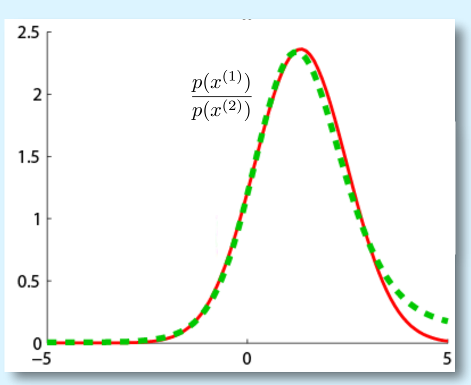
For some models, we only have access to an unnormalised probability or partial knowledge of the distribution.

***Basic idea:***
Transform density ratio estimation into class probability estimation




*We compare the estimated distribution to the true distribution using samples.*




**Learning principle: Two-sample tests**

$$\frac{p(\hat{\mathbf{x}})}{p(\tilde{\mathbf{x}})} = 1 \qquad p(\hat{\mathbf{x}}) = p(\tilde{\mathbf{x}})$$

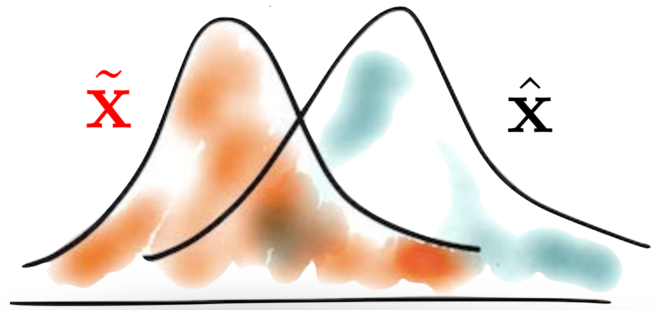
*Interest is not in estimating the marginal probabilities, only in how they are related.*

# Bayesian Two-sample Testing

Combine data $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\} = \{\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_{\hat{n}}, \tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_{\tilde{n}}\}$

Assign labels $\{y_1, \ldots, y_N\} = \{+1, \ldots, +1, -1, \ldots, -1\}$

Equivalence $p(\hat{\mathbf{x}}) = p(\mathbf{x}|y = +1) \qquad p(\tilde{\mathbf{x}}) = p(\mathbf{x}|y = -1)$




Density Ratio $\frac{p(\hat{\mathbf{x}})}{p(\tilde{\mathbf{x}})}$

Bayes' Rule $p(\mathbf{x}|y) = \frac{p(y|\mathbf{x})p(\mathbf{x})}{p(y)}$

Conditional

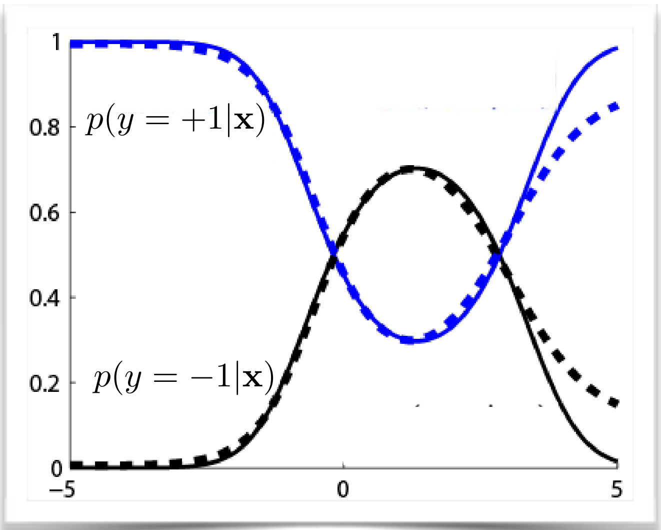
$$\frac{p(\hat{\mathbf{x}})}{p(\tilde{\mathbf{x}})} = \frac{p(\mathbf{x}|y = 1)}{p(\mathbf{x}|y = -1)}$$

Bayes' Subst.

$$= \frac{p(y = +1|\mathbf{x})p(\mathbf{x})}{p(y = +1)} \bigg/ \frac{p(y = -1|\mathbf{x})p(\mathbf{x})}{p(y = -1)}$$

Class probability

$$\frac{p(\hat{\mathbf{x}})}{p(\tilde{\mathbf{x}})} = \frac{p(y = +1|\mathbf{x})}{p(y = -1|\mathbf{x})}$$




***Computing a density ratio is equivalent to class probability estimation.***

# Testing to Adversarial Learning

Scoring Function $p(y=+1|\mathbf{x}) = D_\theta(\mathbf{x})$ $\quad p(y=-1|\mathbf{x}) = 1 - D_\theta(\mathbf{x})$

Bernoulli outcome $\log p(y|\mathbf{x}) = \log D_\theta(\hat{\mathbf{x}}) + \log(1 - D_\theta(\tilde{\mathbf{x}}))$

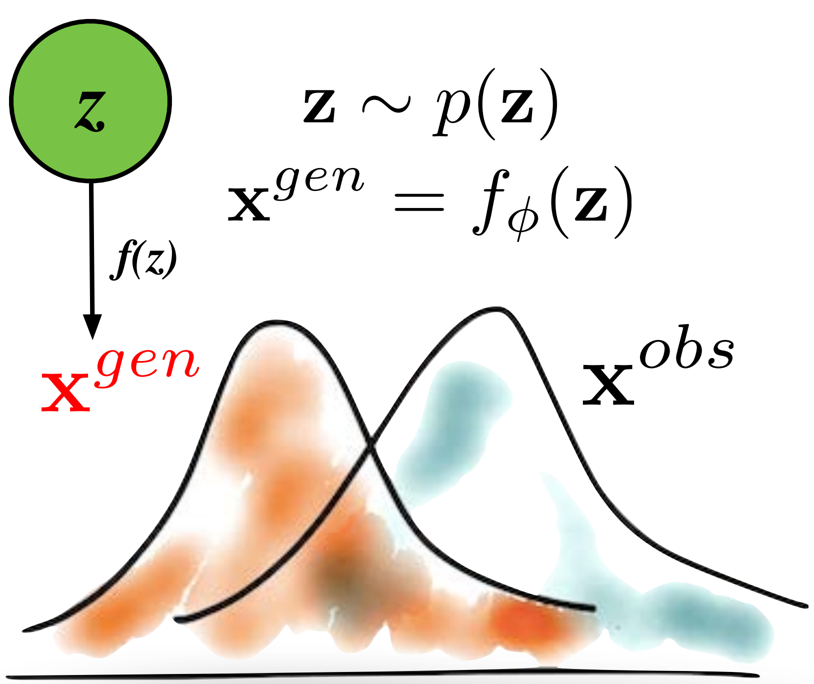
**Two-sample criterion** $\mathcal{F}(\mathbf{x}, \theta) = \mathbb{E}_{p(x^{obs})}[\log D_\theta(\mathbf{x}^{obs})] + \mathbb{E}_{p(x^{gen})}[\log(1 - D_\theta(\mathbf{x}^{gen}))]$

---

## *Generative Adversarial Networks*

$$\boxed{\mathcal{F}(\mathbf{x}, \theta, \phi) = \mathbb{E}_{p(x^{obs})}[\log D_\theta(\mathbf{x}^{obs})] + \mathbb{E}_{p(z)}[\log(1 - D_\theta(f_\phi(\mathbf{z})))]}$$

$$\mathbf{z} \sim p(\mathbf{z})$$
$$\mathbf{x}^{gen} = f_\phi(\mathbf{z})$$




Alternating optimisation

$$\min_\phi \max_\theta \mathcal{F}(\mathbf{x}, \theta, \phi)$$

**Instances of testing and inference:**

- Two-sample density ratio estimation
- Importance estimation
- Noise-contrastive estimation
- Adversarial learning

# Part IV

# Tools for Algorithm Building

Tools for constructing scalable algorithms

- Amortised inference
- Stochastic optimisation

# Variational EM

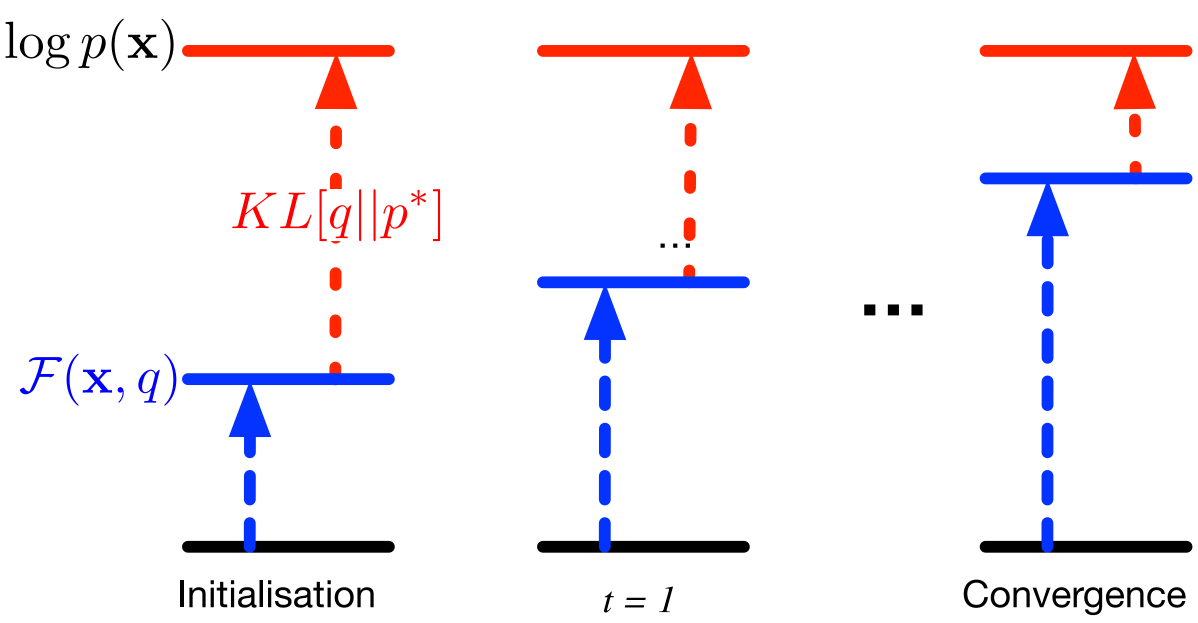
$$\mathcal{F}(\mathbf{x}, q) = \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}|\mathbf{z})] - KL[q(\mathbf{z})\|p(\mathbf{z})]$$

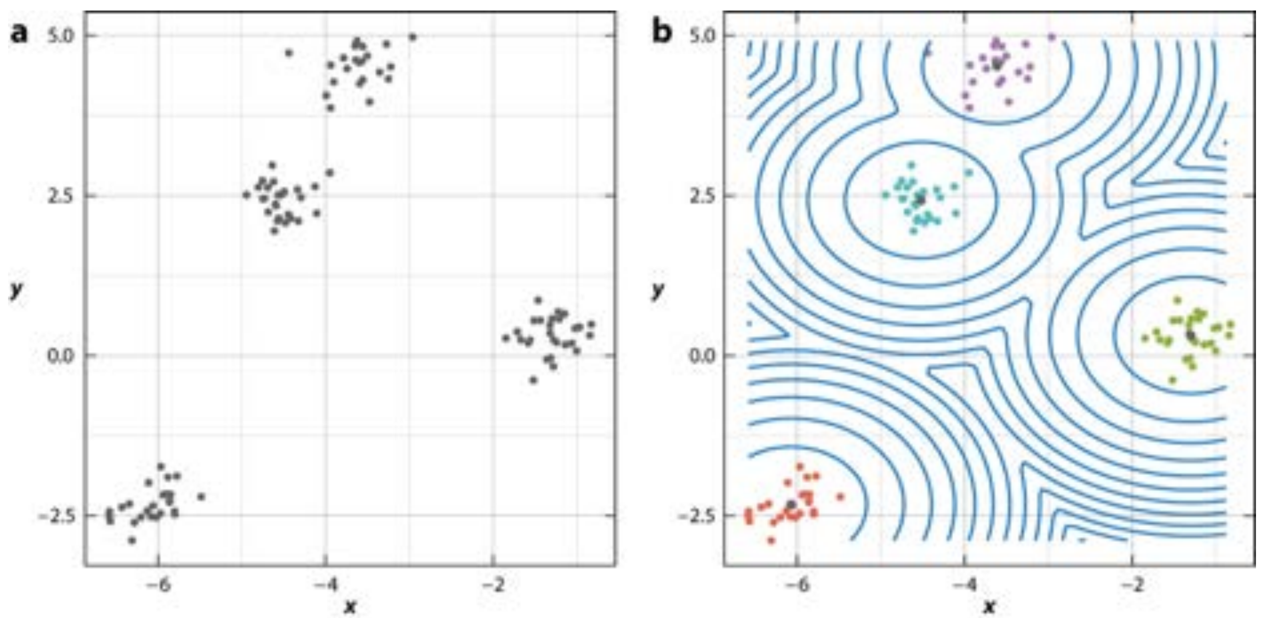
**Alternating optimisation** for the variational parameters and then model parameters (VEM).

**Repeat:**

E-step $\phi \propto \nabla_\phi \mathcal{F}(\mathbf{x}, q)$ ***Var. params***

M-step $\theta \propto \nabla_\theta \mathcal{F}(\mathbf{x}, q)$ ***Model params***



# Stochastic Approximation

$$\mathcal{F}(\mathbf{x}, q) = \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}|\mathbf{z})] - KL[q(\mathbf{z})\|p(\mathbf{z})]$$

Optimise using a a **stochastic gradient based on a mini-batch** of data.
Many names: *online EM, stochastic approximation EM, stochastic variational inference*.

**Repeat:**

E-step (compute *q*) **(Inference)**

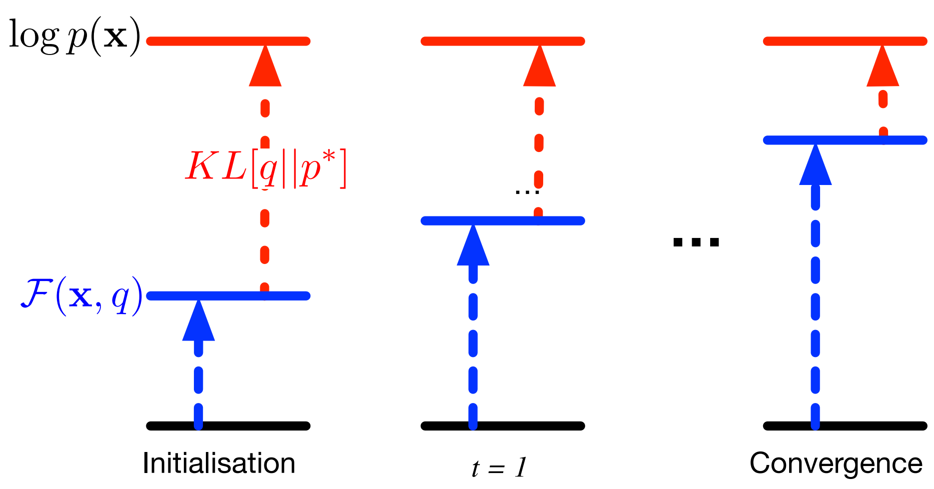
*For i = 1, ... N*

$$\phi_n \propto \nabla_\phi \mathbb{E}_{q_\phi(z)}[\log p_\theta(\mathbf{x}_n|z_n)] - \nabla_\phi KL[q(z_n)\|p(z)]$$

***N*** **is a mini-batch:** sampled with replacement from the full data set or received online.

M-step **(Parameter Learning)**

$$\theta \propto \boxed{\frac{1}{N}\sum_n} \mathbb{E}_{q_\phi(z)}[\nabla_\theta \log p_\theta(\mathbf{x}_n|z_n)]$$



# Memoryless Inference

***E-step does not reuse any previous computation.***

**Repeat:**

E-step (compute *q*) **(Inference)**

*For i = 1, … N*

$$\phi_n \propto \nabla_\phi \mathbb{E}_{q_\phi(z)}[\log p_\theta(\mathbf{x}_n|z_n)] - \nabla_\phi KL[q(z_n)||p(z)]$$

M-step **(Parameter Learning)**

$$\theta \propto \frac{1}{N}\sum_n \mathbb{E}_{q_\phi(z)}[\nabla_\theta \log p_\theta(\mathbf{x}_n|z_n)]$$

*Memoryless*: Any inference computations are discarded after the M-step update

# Amortised Inference

**Repeat:**

E-step (compute $q$)

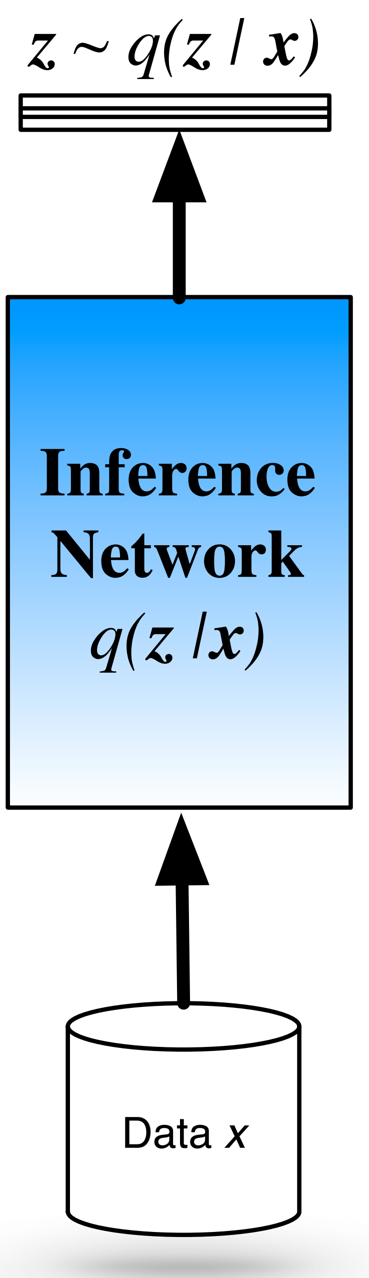
*For i = 1, ... N*

$$\phi_n \propto \nabla_\phi \mathbb{E}_{q_\phi(z)}[\log p_\theta(\mathbf{x}_n|z_n)] - \nabla_\phi KL[q(z_n)\|p(z)]$$

Instead of solving for every observation, amortise using a model.

M-step

$$\theta \propto \frac{1}{N}\sum_n \mathbb{E}_{q_\phi(z)}[\nabla_\theta \log p_\theta(\mathbf{x}_n|z_n)]$$




- **Inference network**: $q$ is an ***encoder,*** an ***inverse*** model, ***recognition model***.
- Parameters of $q$ are now a set of *global parameters* used for inference of all data points - test and train.
- **Amortise (spread) the cost of inference over all data.**
- Joint optimisation of variational and model parameters.

Inference networks provide an efficient mechanism for **posterior inference with memory**

# Amortised Variational Inference

$$\mathcal{F}(\mathbf{x}, q) = \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}|\mathbf{z})] - KL[q(\mathbf{z})\|p(\mathbf{z})]$$

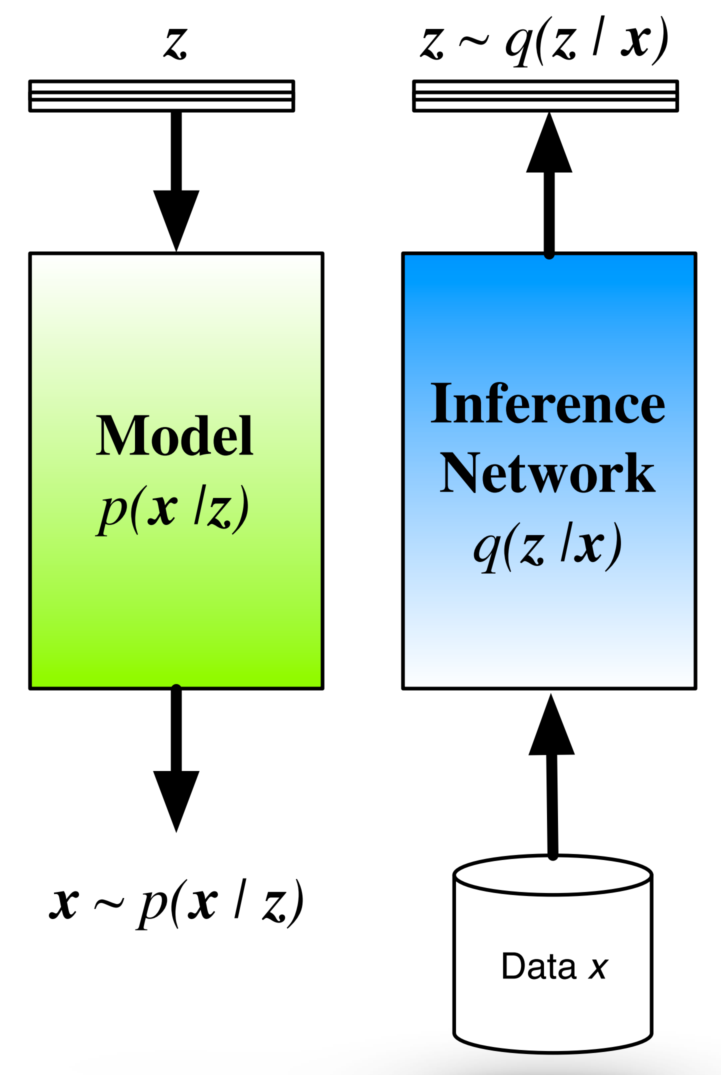
Approx. Posterior

Reconstruction

Penalty

*Stochastic encoder-decoder system to implement variational inference.*

- **Model (Decoder):** likelihood $p(x|z)$.
- **Inference (Encoder):** variational distribution $q(z|x)$
- Transforms an auto-encoder into a generative model




Specific combination of variational inference in latent variable models using inference networks
Variational Auto-encoder

*But don't forget what your model is, and what inference you use.*

# Minimum Description Length

$$\mathcal{F}(\mathbf{x}, q) = \boxed{\mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}|\mathbf{z})]} - \boxed{KL[q(\mathbf{z})\|p(\mathbf{z})]}$$

Stochastic encoder

Data code-length

Hypothesis code

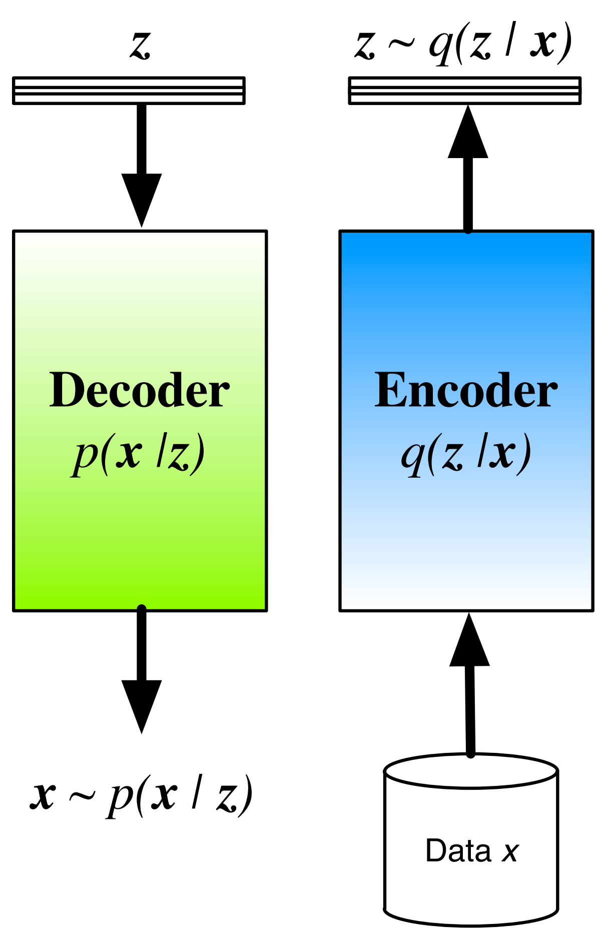
***Stochastic encoder-decoder systems implement amortised variational inference.***

Regularity in our data that can be explained with latent variables, implies that the data is ***compressible***.

*Minimum Description Length (MDL):*
*Inference is a problem of Compression.*

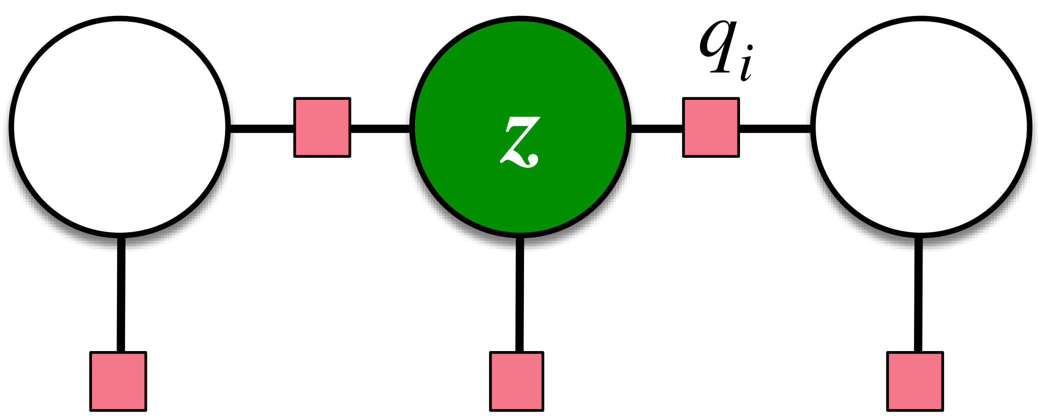
we must find the ideal shortest message of our data *x:* marginal likelihood.

- Must introduce an approximation to the ideal message.
- **Encoder:** variational distribution $q(z|x)$,
- **Decoder:** likelihood $p(x|z)$.



# Amortised Message Passing

Expectation Propagation




Factorised assumption

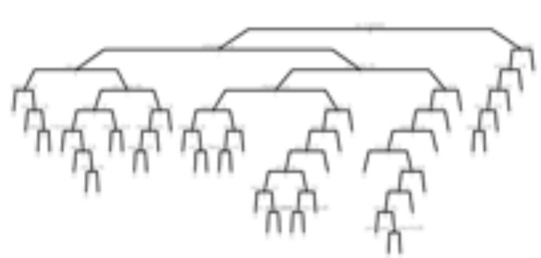
$$p(z|\mathcal{D}) = \prod_i f_i(z)$$
$$\approx \prod_i q_i(z) = q(z)$$

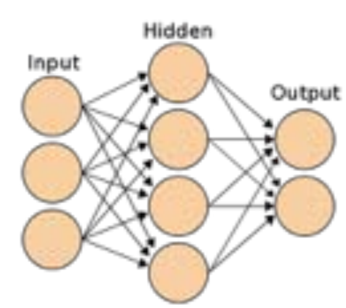
***Memoryless inference:*** *solve and update cavity distributions iteratively.*

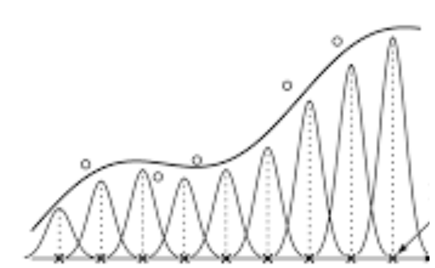
$$q_i = \arg\min_{q \in \mathcal{Q}} D_{KL}[f^i q^{\setminus i} \| q^i q^{\setminus i}]$$

*Amortised inference:* *Use a model (trees, deep nets, basis functions).*

$$q_i = h(\{q^i\}, \mathcal{D}; \theta)$$





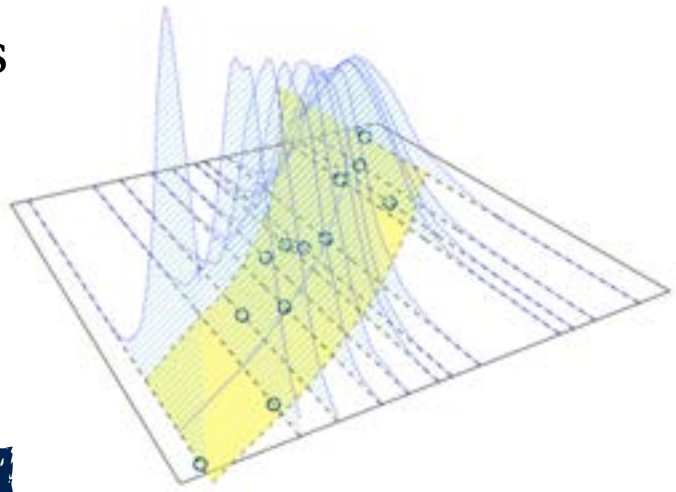



# Amortised Predictive Distributions

Bayesian Dark Knowledge

Posterior predictive distributions in Bayesian neural networks

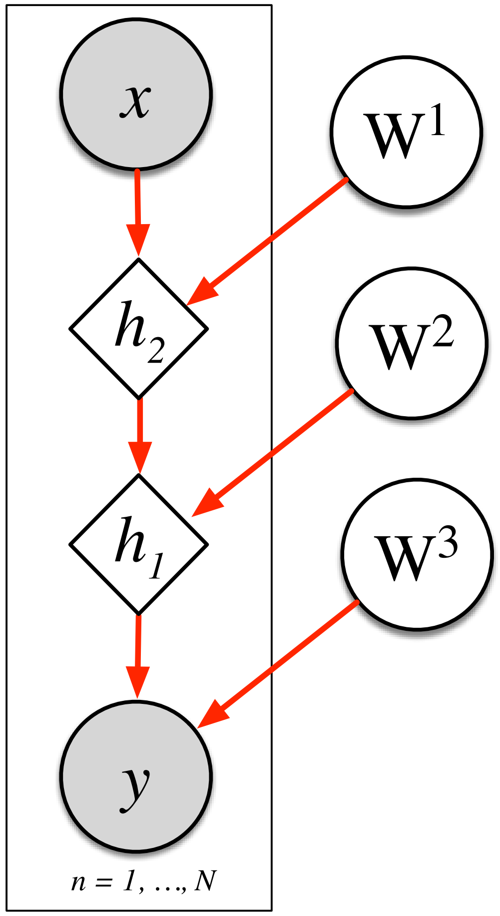
$$p(y^*|x^*, X, Y) = \int p(y^*|x^*, W)p(W|X, Y)dW$$



**Memoryless prediction:** *compute by Monte Carlo*

$$W^{\{s\}} \sim p(W|X, Y)$$

$$q(y^*|x^*) = \frac{1}{S}\sum_{s=1}^{S} p(y^*|x^*, W^{(s)})$$




*Amortised predictions:*
*distillation using a deep network.*

$$p(y^*|x^*, X, Y) = f(x^*, \theta)$$

# Stochastic Optimisation

*Common gradient problem*

$$\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{z})}[f_\theta(\mathbf{z})] = \nabla \int \boxed{q_\phi(\mathbf{z})} \boxed{f_\theta(\mathbf{z})} d\mathbf{z}$$

- Don't know this expectation in general.
- Gradient is of the parameters of the distribution w.r.t. which the expectation is taken.

Two general approaches:

- **Deterministic methods:** use additional bounds to simplify computation - local variational methods.
- **Stochastic methods:** Compute the expectation by Monte Carlo and exploit properties of the distributions.

***Typical problem areas:***

- Generative models and inference
- Reinforcement learning and control
- Operations research and inventory control
- Monte Carlo simulation
- Finance and asset pricing

1. *Pathwise estimator*: Differentiate the function *f(z)*
2. *Score-function estimator*: Differentiate the density *q(z|x)*

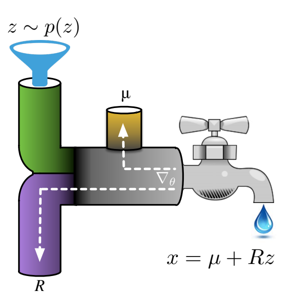
# Stochastic Gradient Estimators

$$\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{z})}[f_\theta(\mathbf{z})] = \nabla \int \boxed{q_\phi(\mathbf{z})} \boxed{f_\theta(\mathbf{z})} d\mathbf{z}$$

## Pathwise Estimator

When easy to use transformation is available and differentiable function *f*.

$$= \mathbb{E}_{p(\epsilon)}[\nabla_\phi f_\theta(g(\epsilon, \phi))]$$




$$z \sim q_\phi(\mathbf{z})$$

$$\mathbf{z} = g(\epsilon, \phi) \quad \epsilon \sim p(\epsilon)$$

*Other names:*

Stochastic backpropagation
Perturbation analysis
Reparameterisation trick
Affine-independent inference

## Score-function estimator

When function *f* non-differentiable and *q(z)* is easy to sample from.

$$= \mathbb{E}_{q(z)}[f_\theta(\mathbf{z}) \nabla_\phi \log q_\phi(\mathbf{z})]$$

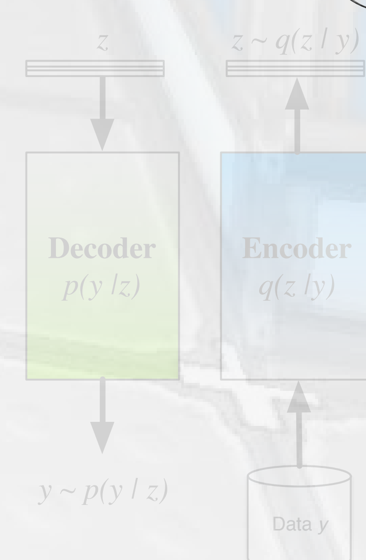
*Other names:*

Likelihood ratio method
REINFORCE and policy gradients
Automated inference
Black-box inference

***Doubly stochastic estimators***

Part V

# The Case of Variational Auto-encoders




Explore different types of VAEs

- Discrete and continuous latents
- Static, sequential, volumetric.
- Differentiable and non-differentiable fns.

# Variational Auto-encoders in General

**Variational Auto-encoder (VAE)**
Amortised variational inference for latent variable models

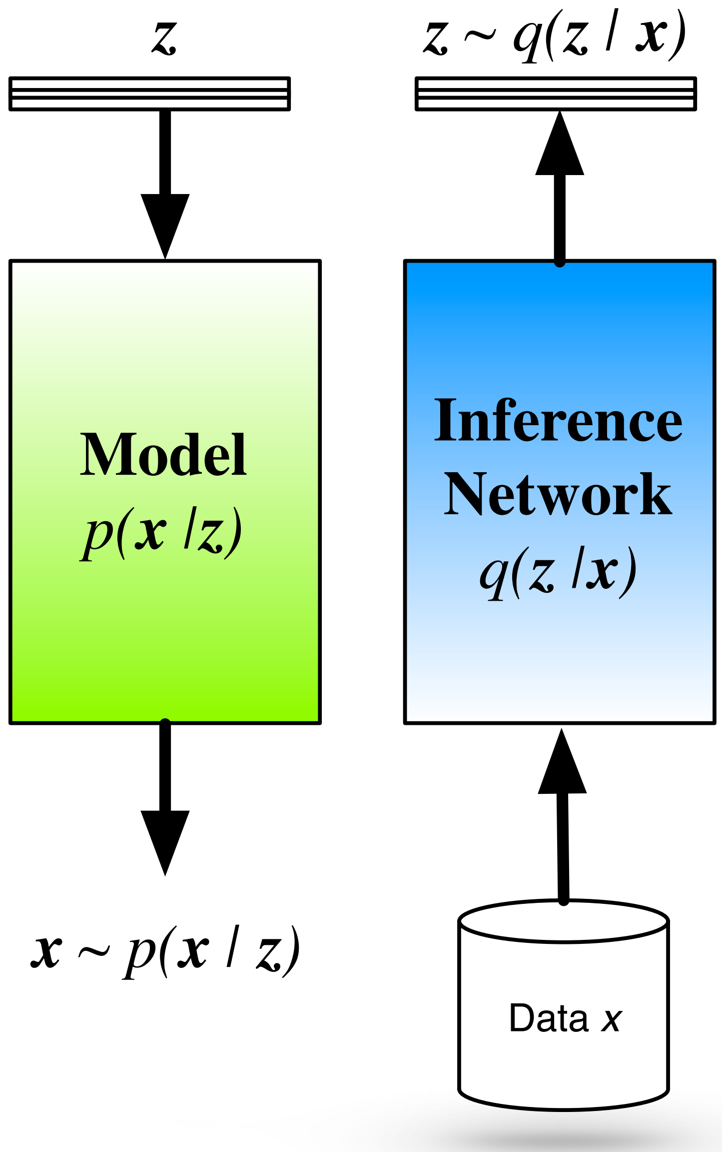
$$\mathcal{F}(q) = \mathbb{E}_{q_\phi(z)}[\log p_\theta(\mathbf{x}|\mathbf{z})] - KL[q_\phi(\mathbf{z}|\mathbf{x}) \| p(\mathbf{z})]$$

***Design choices***

- ***Prior on the latent variable***
  - Continuous, Discrete, Gaussian, Bernoulli, Mixture
- ***Likelihood function***
  - iid (static), sequential, temporal, spatial
- ***Approximating posterior***
  - distribution, sequential, spatial

***For scalability and ease of implementation***

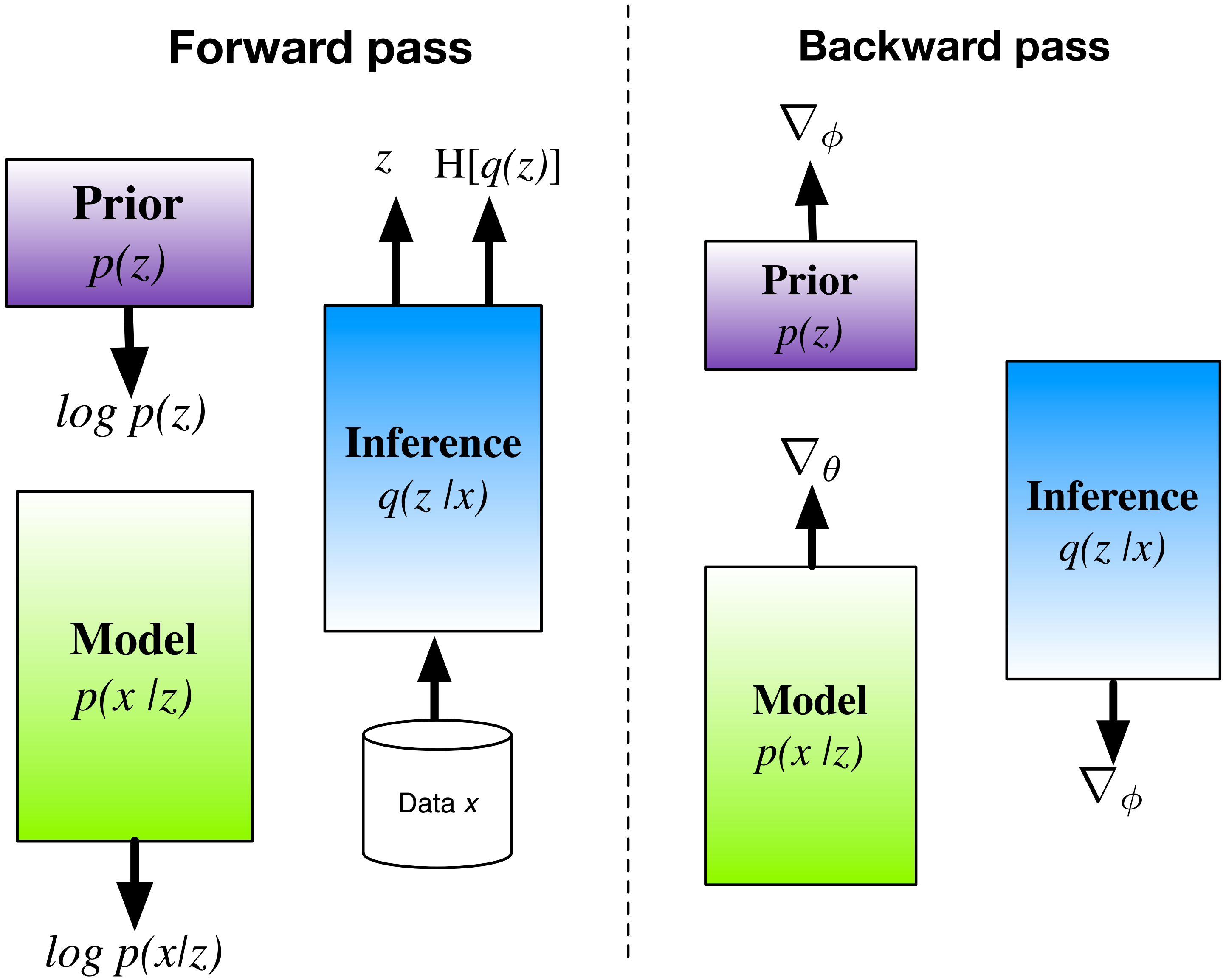
- Stochastic gradient descent (and variants),
- stochastic gradient estimation



# Implementing a Variational Algorithm

Variational inference turns integration into optimisation: **Automated Tools:**

- **Differentiation:** Theano, Torch7, TensorFlow, Stan.
- **Message passing:** infer.NET

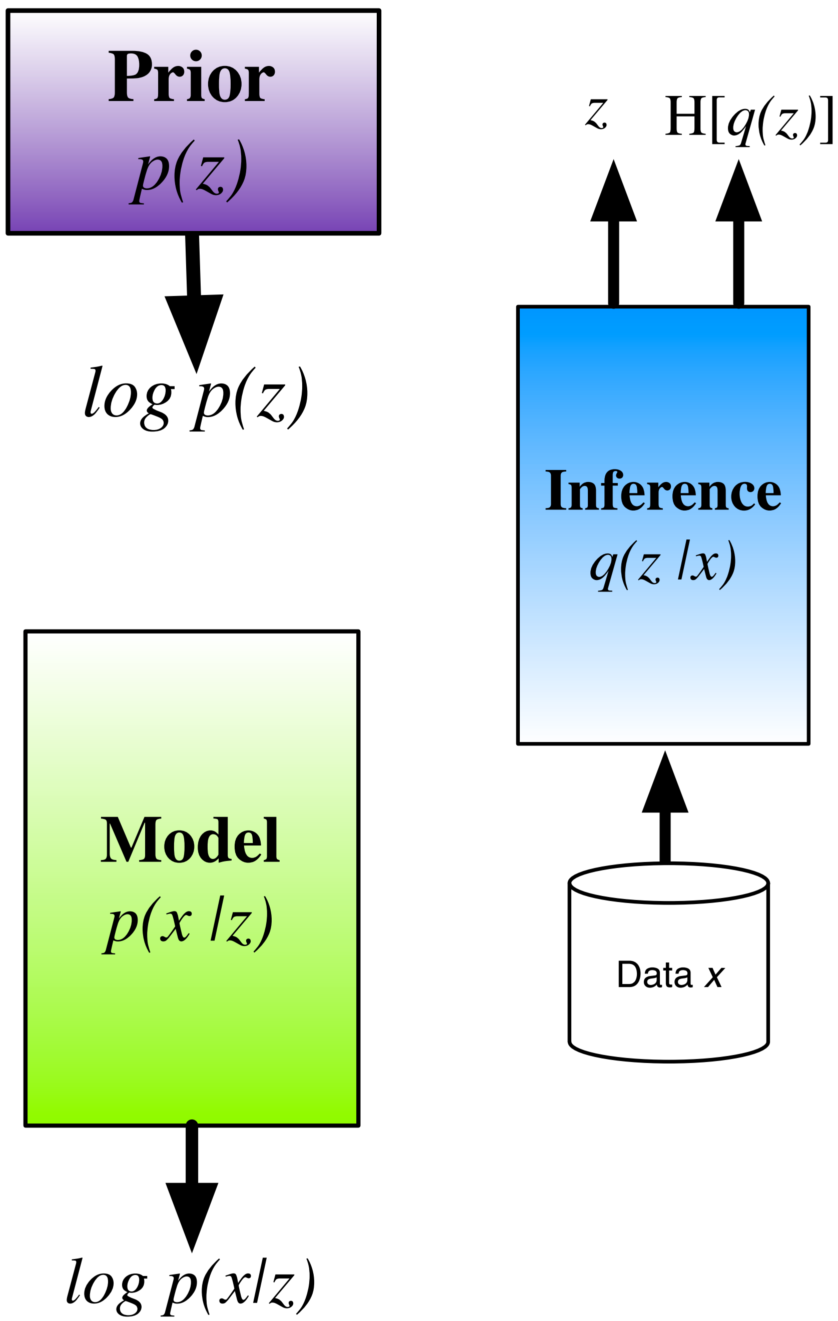



- Stochastic gradient descent and other preconditioned optimisation.
- Same code can run on both GPUs or on distributed clusters.
- Probabilistic models are modular, can easily be combined.

*Ideally want probabilistic programming using variational inference.*

# Latent Gaussian VAE

Deep Latent Gaussian Model




$$p(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathbf{I})$$

$$p(\mathbf{x}|f_\theta^p(\mathbf{z}))$$

$$p_\theta(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mu_\theta^p(\mathbf{z}), \Sigma_\theta^p(\mathbf{z}))$$

$$q_\phi(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mu_\phi^q(\mathbf{x}), \Sigma_\phi^q(\mathbf{x}))$$

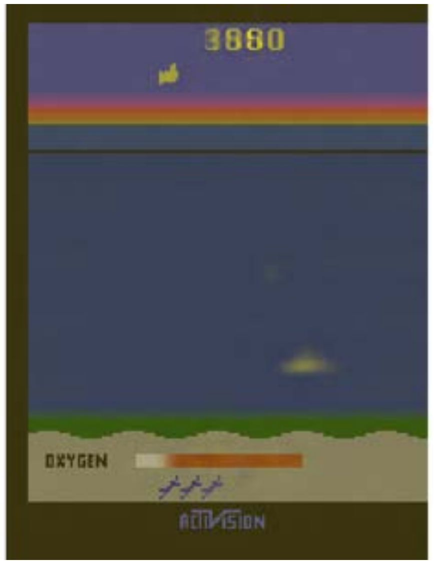
$$\mathcal{F}(\mathbf{x}, q) = \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}|\mathbf{z})] - KL[q(\mathbf{z})\|p(\mathbf{z})]$$

*All functions are deep networks.*

# Latent Gaussian VAE

*Latent space disentangles the input data*

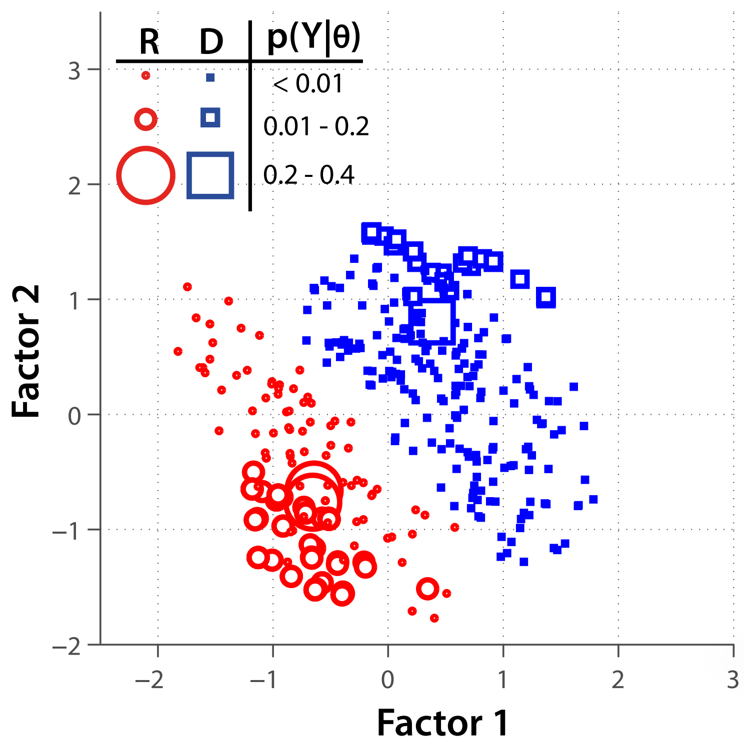
Oxygen/Swimmers



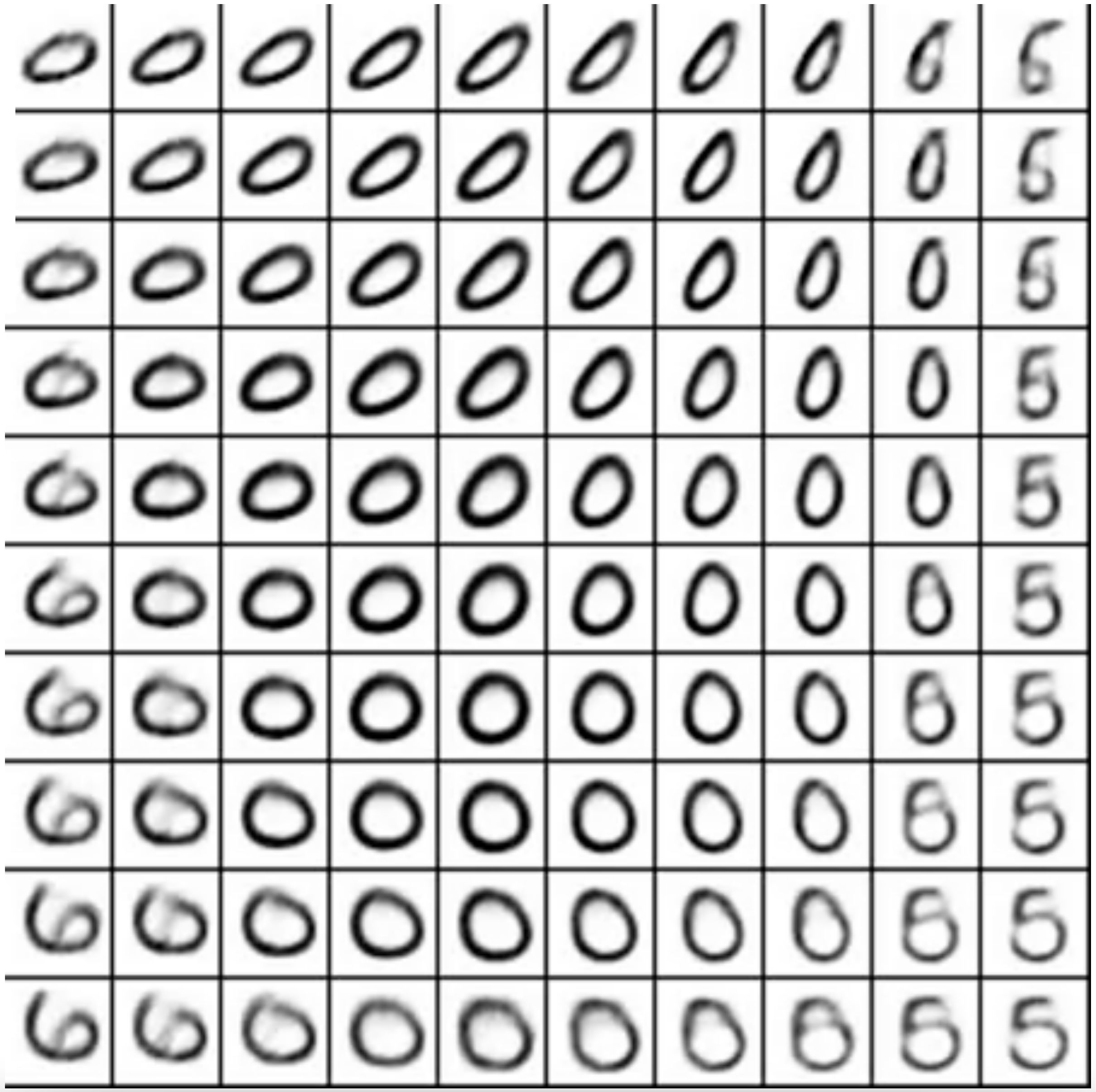

Moving Left


*Latent space and likelihood bound gives a visualisation of importance.*

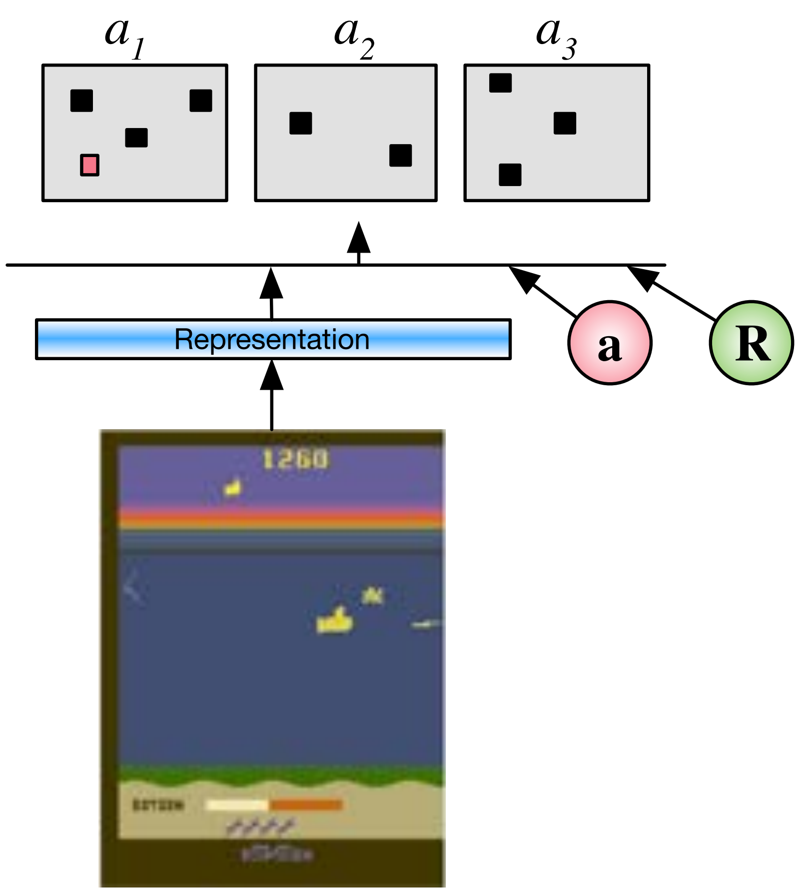
**Latent Factor Embedding**

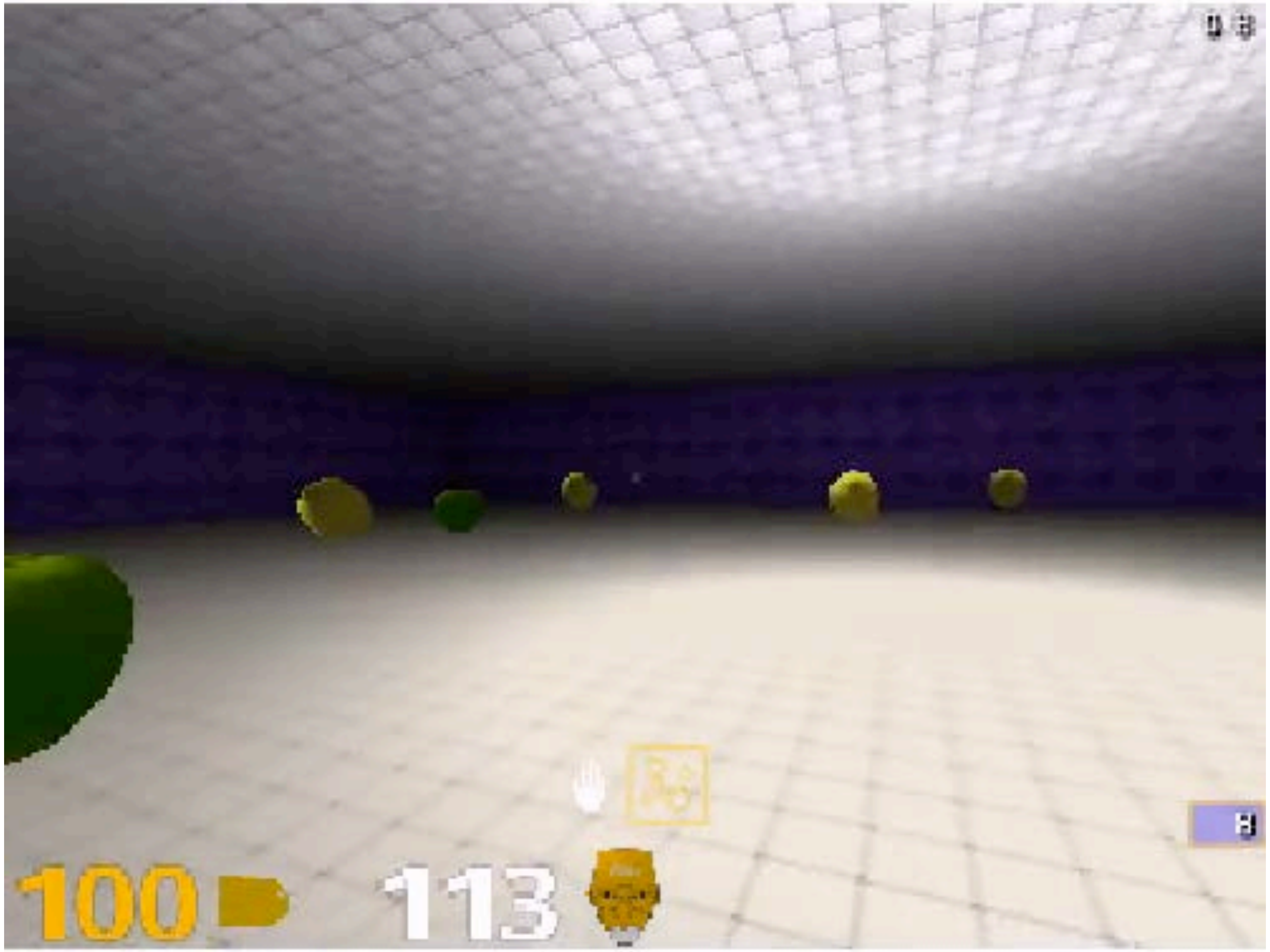



*3 dimensional latent variable of MNIST*



# VAE Representations

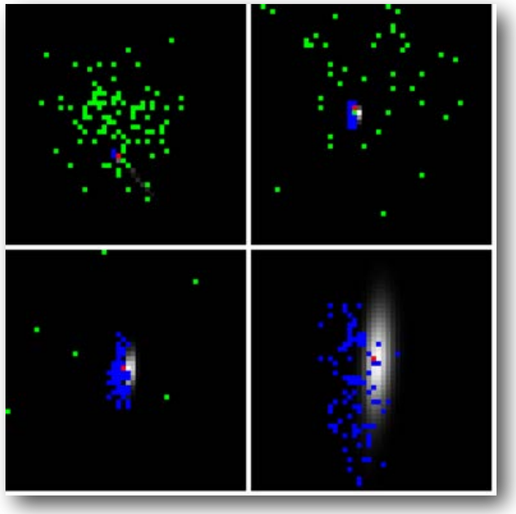


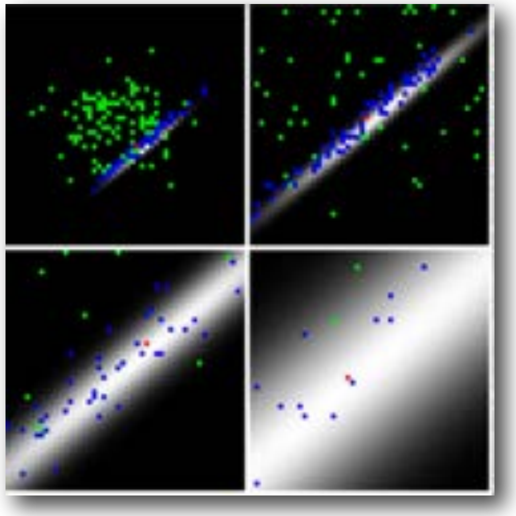



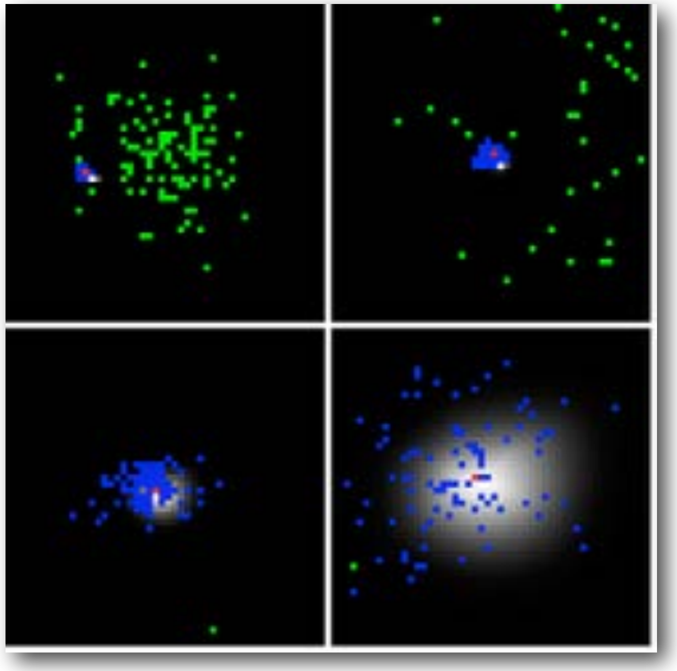

*Representations are useful for strategies such as episodic control.*

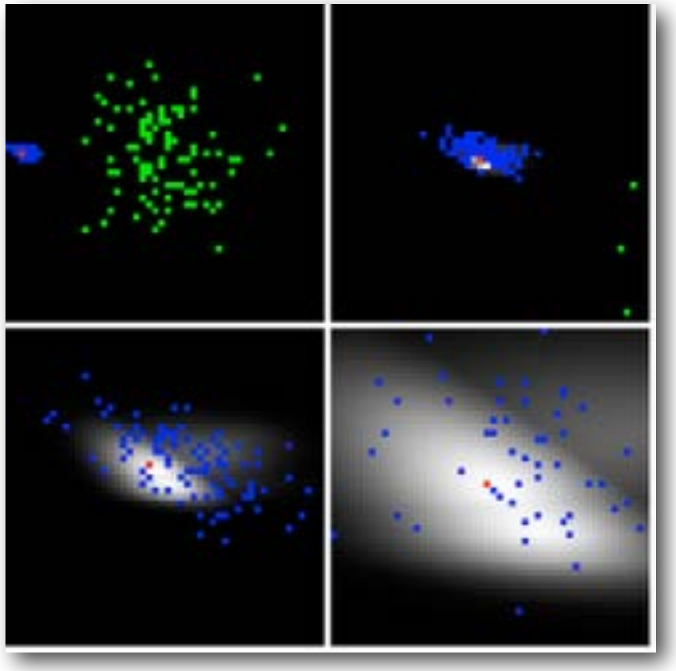
# Latent Gaussian VAE

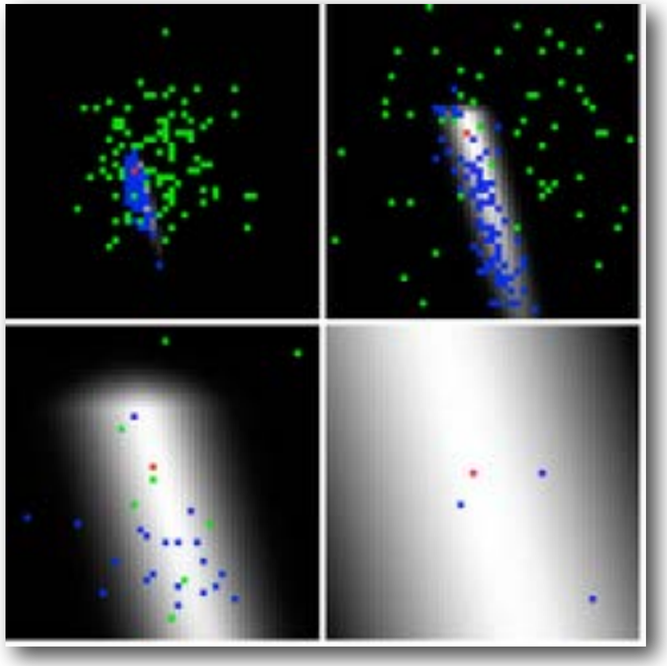
*Require flexible approximations for the types of posteriors we are likely to see.*





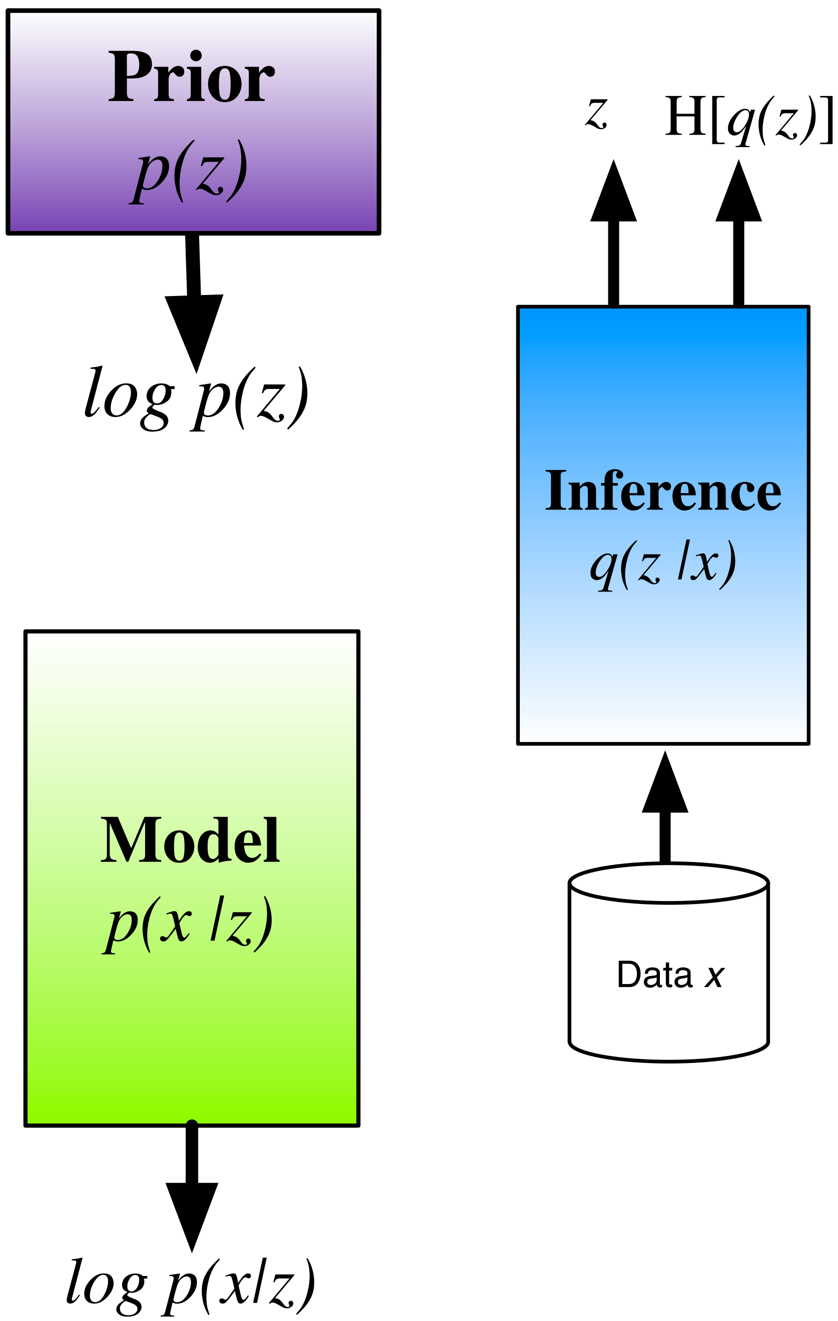






# Latent Binary VAE

Deep Auto-regressive Networks



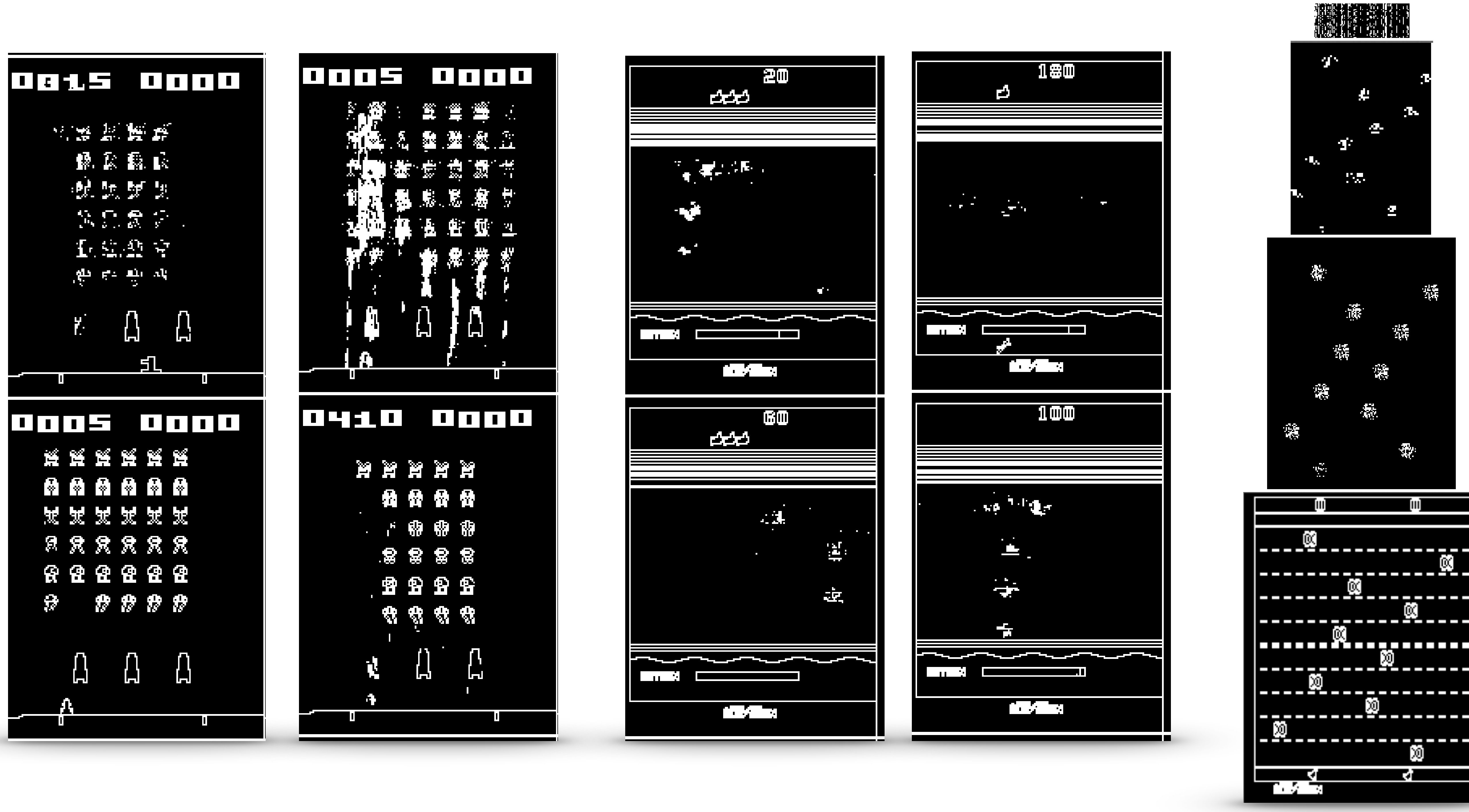

$$p(z_i|\mathbf{z}_{<i}) = Bern(z_i|f(\mathbf{z}_{<i}))$$
$$p(\mathbf{z}) = \prod_i p(z_i|\mathbf{z}_{<i})$$

$$p(\mathbf{x}|\mathbf{z}) = \prod_i p(x_i|\mathbf{x}_{<i}, \mathbf{z})$$
$$p(\mathbf{x}|\mathbf{z}) = \prod_i Bern(x_i|f^p_\theta(\mathbf{x}_{<i}, \mathbf{z}))$$

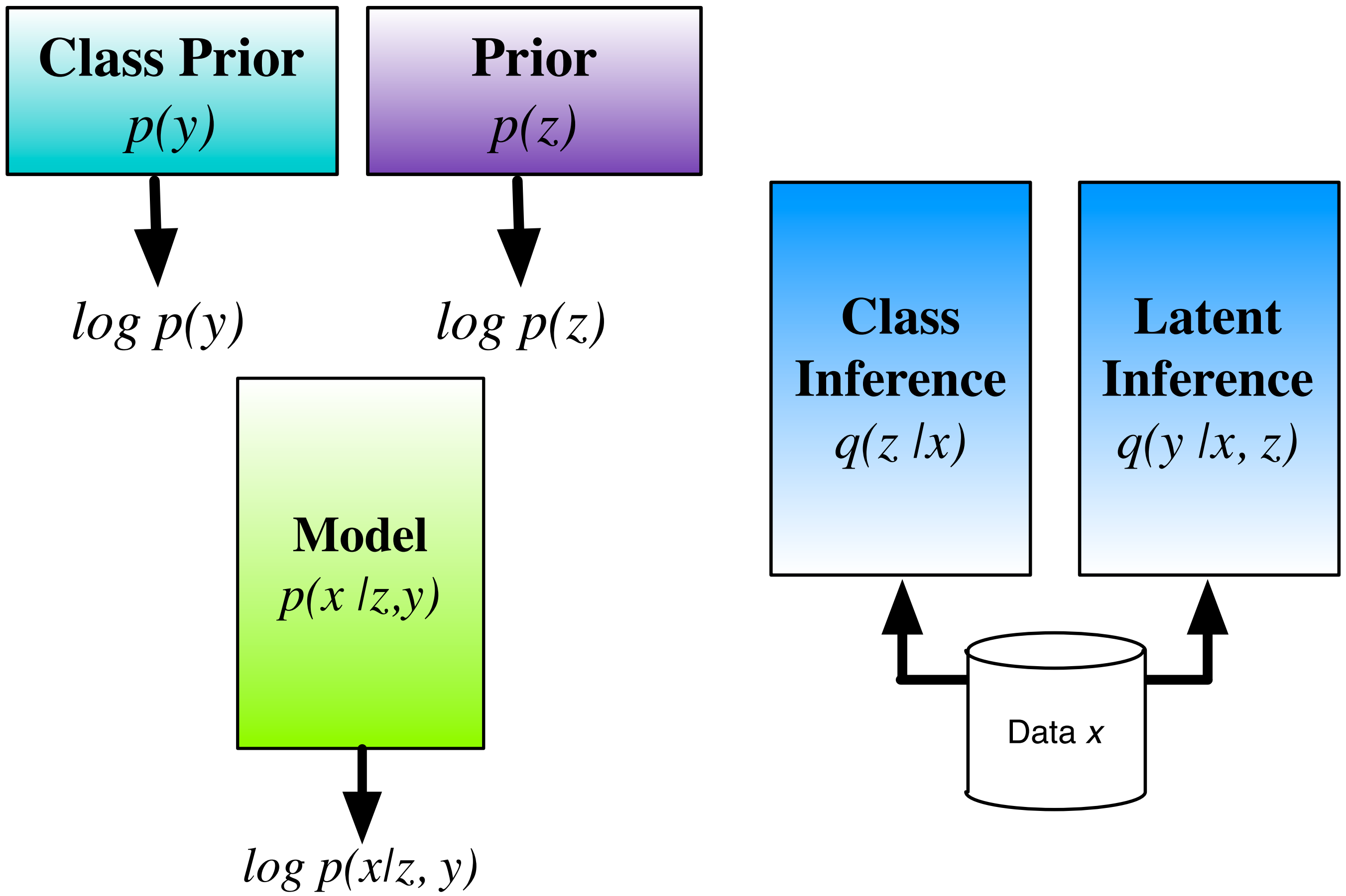
$$q_\phi(\mathbf{z}) = \prod_i q_\phi(z_i|\mathbf{z}_{<i})$$
$$q_\phi(\mathbf{z}) = \prod_i Bern(z_i|f^q_\phi(\mathbf{z}_{<i}))$$

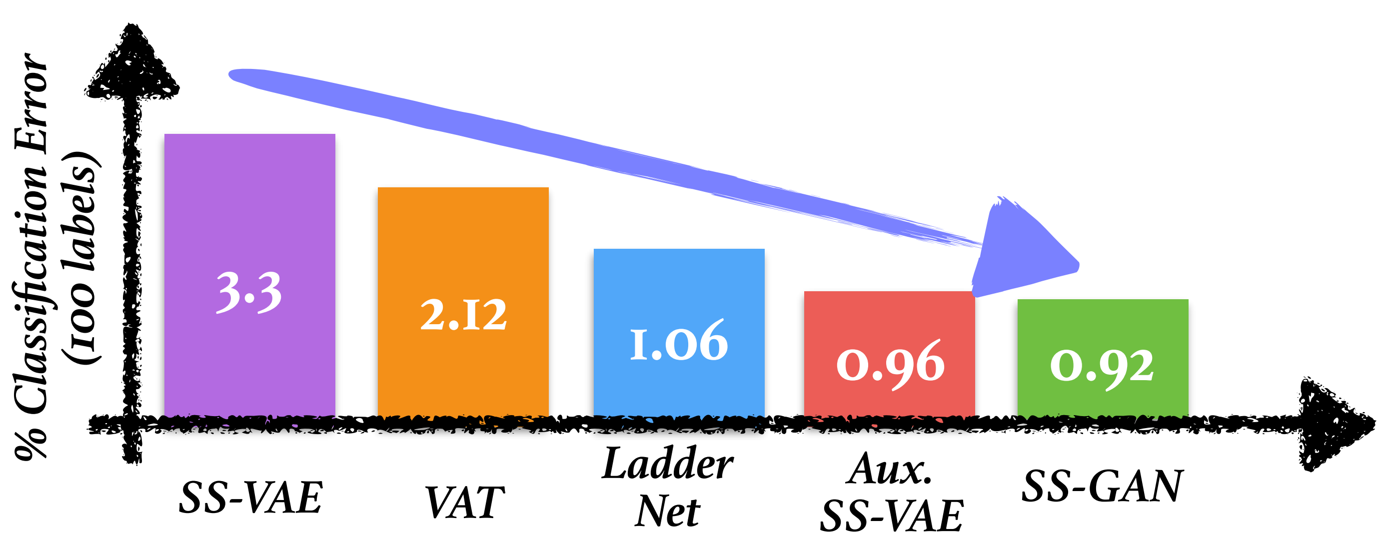
# Latent Binary VAE

*Samples from binarised Atari frames*



# Semi-supervised VAE

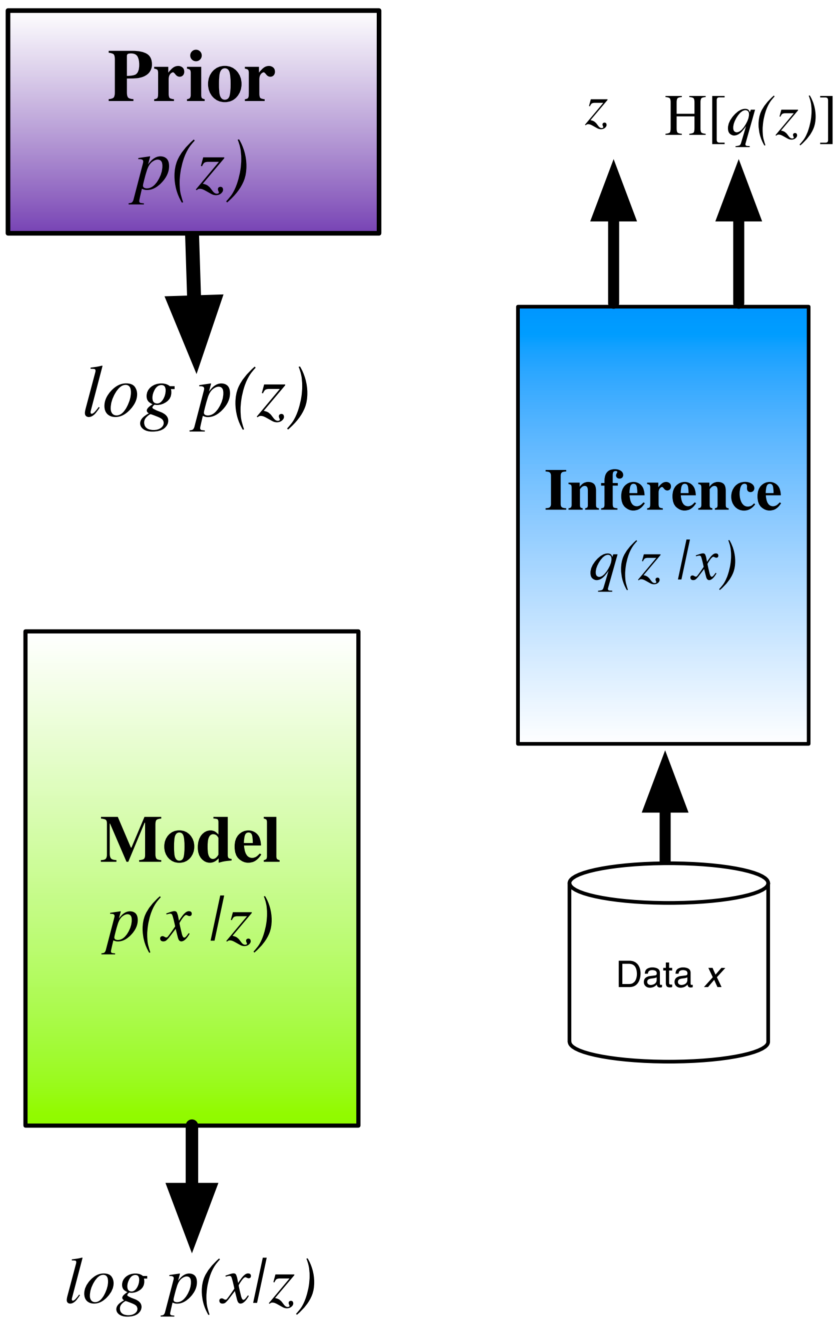



*Visual Analogies*

# Sequential Latent Gaussian VAE

DRAW



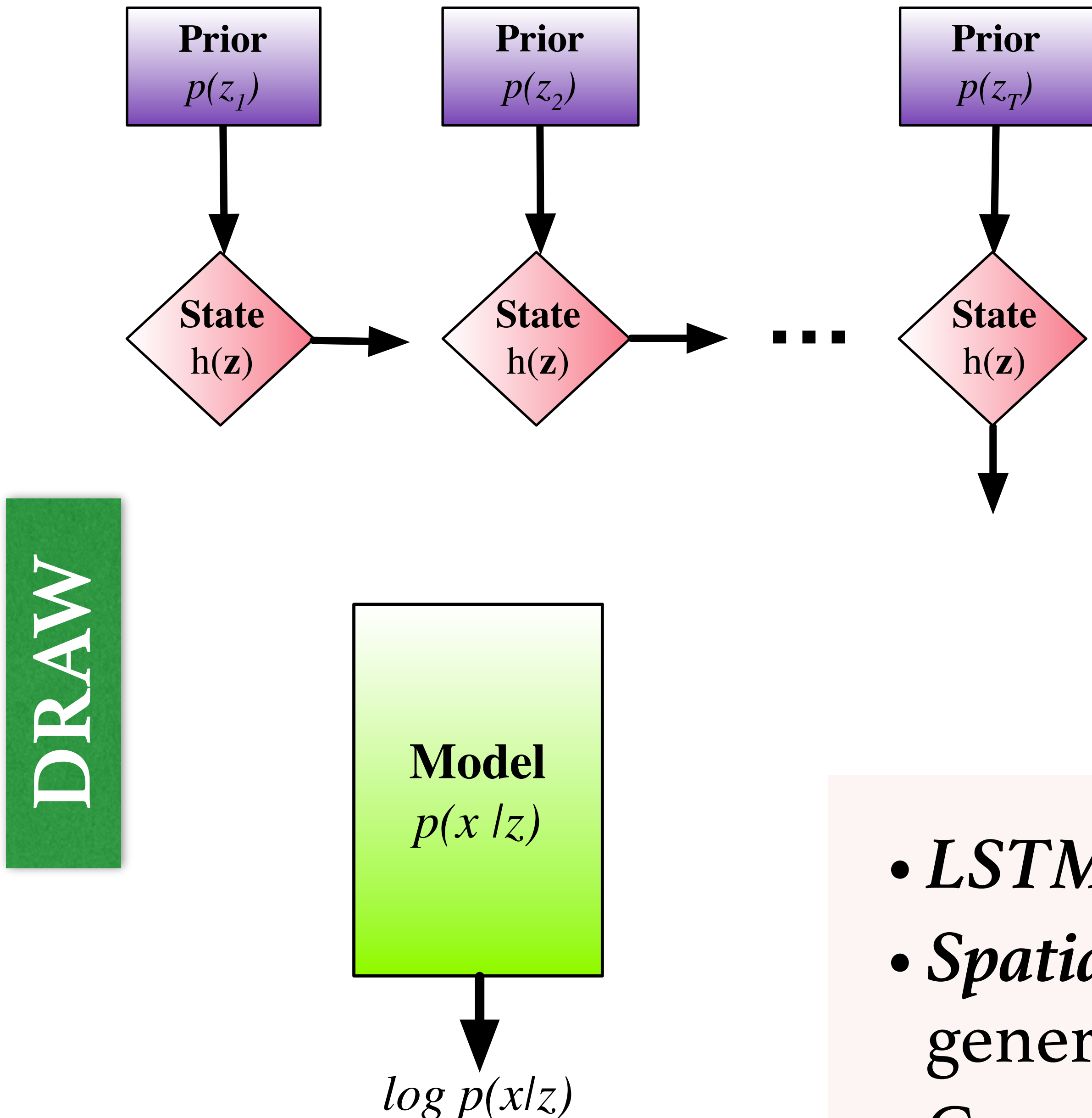

$$p(\mathbf{z}) = \prod_i p(z_i|\mathbf{z}_{<i})$$

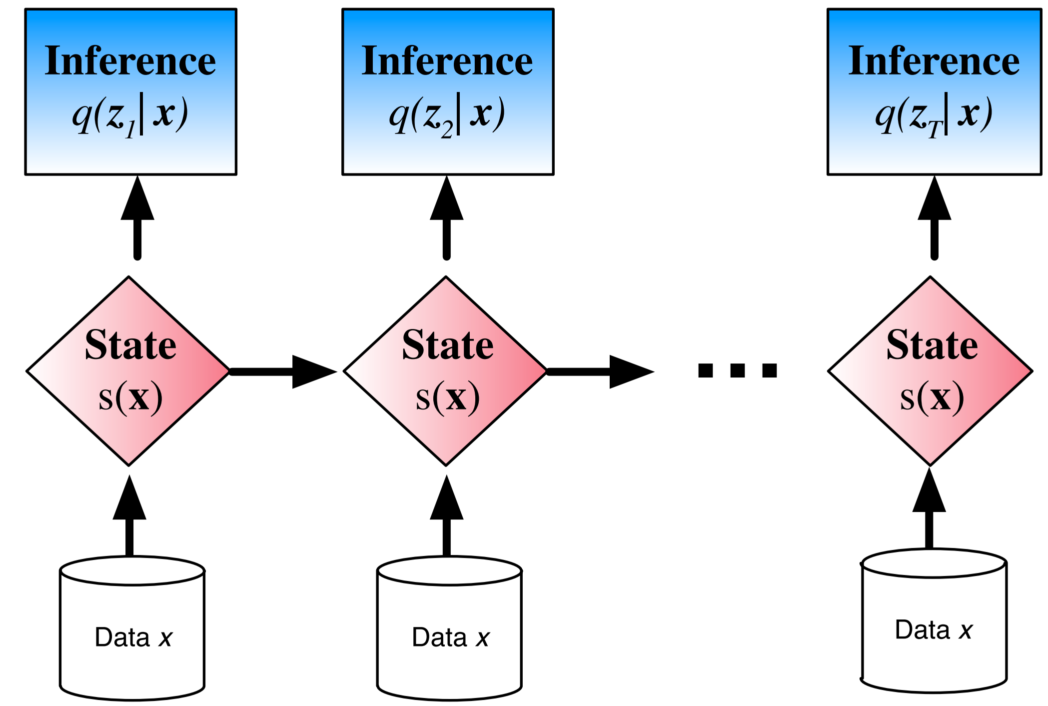
$$p(\mathbf{x}|f_\theta^p(\mathbf{z}))$$

$$p_\theta(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mu_\theta^p(\mathbf{z}), \Sigma_\theta^p(\mathbf{z}))$$

$$q_\phi(\mathbf{z}) = \prod_i q_\phi(z_i|\mathbf{z}_{<i})$$

# Sequential Latent Gaussian VAE



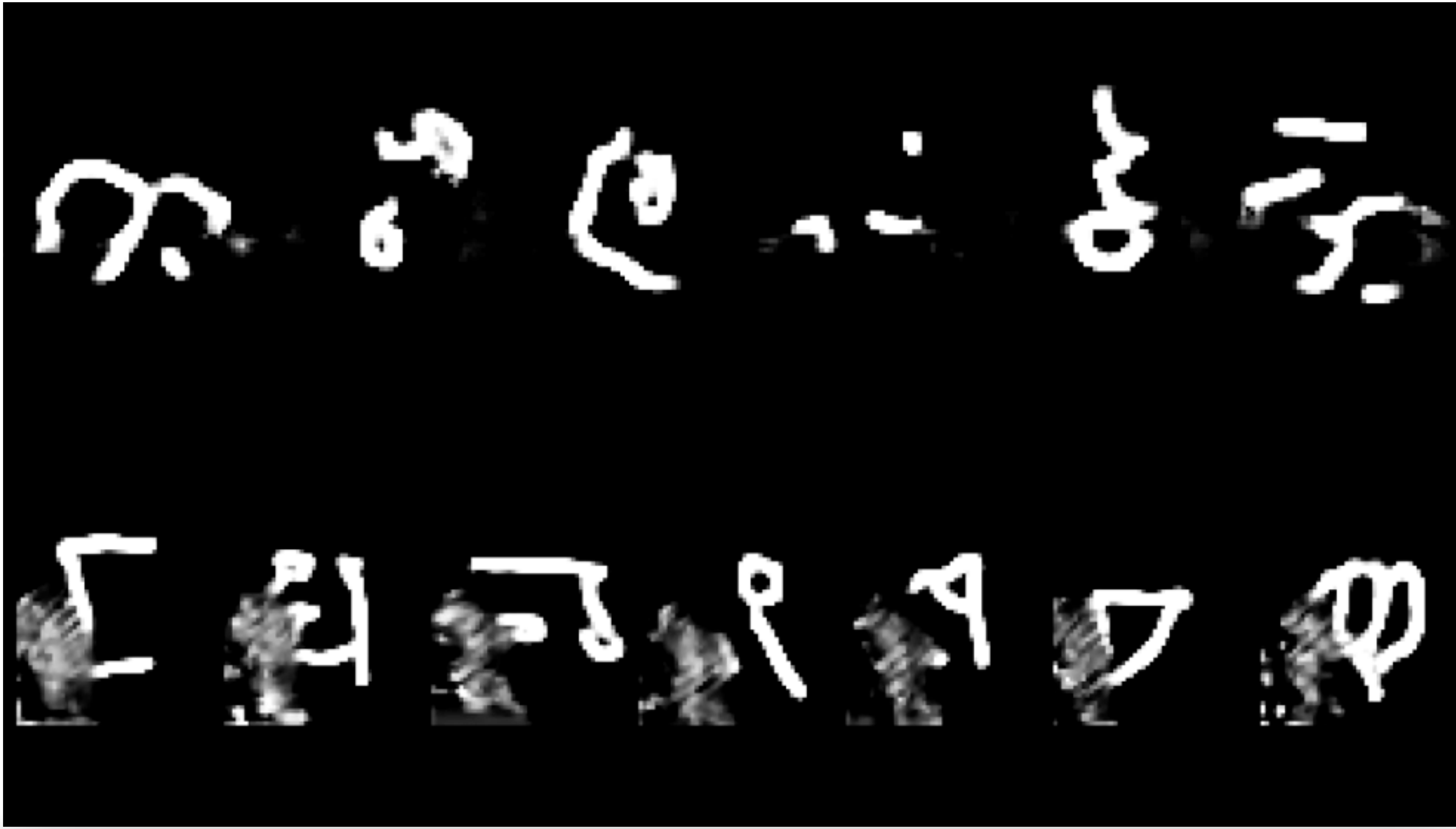




- ***LSTM*** or GRU networks for state modules
- ***Spatial attention*** in both the recognition and generation phase using spatial transformers.
- Can remove inference model RNN and share the generate model state.
- Can include additional canvas

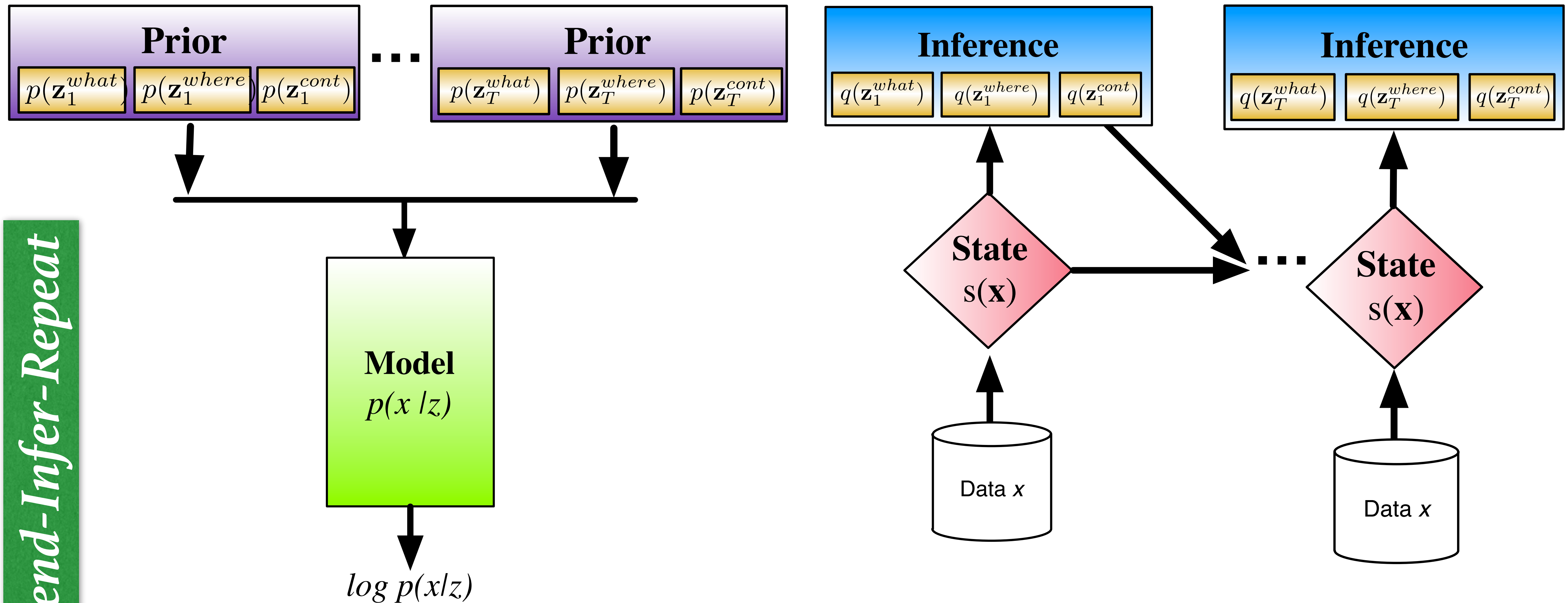
# Sequential Latent Gaussian VAE

DRAW

# Sequential Latent Gaussian VAE



# Structured Sequential VAEs




*AIR - Attend-Infer-Repeat*

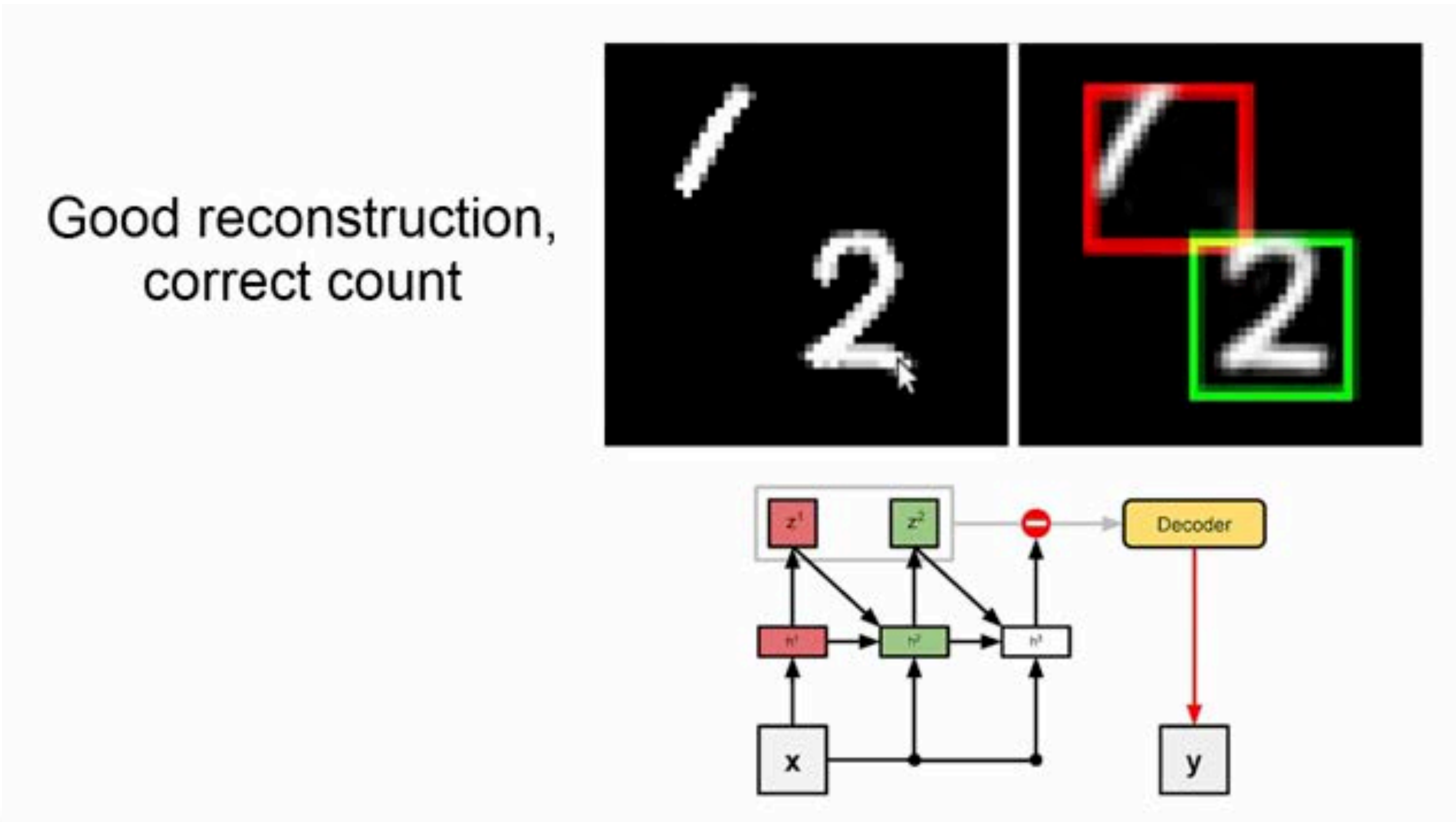
$$p(\mathbf{z}_i) = p(\mathbf{z}_i^{what})p(\mathbf{z}_i^{where})p(\mathbf{z}_i^{cont})$$

$$p(\mathbf{x}|f_\theta^p(\mathbf{z}_1, \ldots, \mathbf{z}_T))$$

$$q(\mathbf{z}_{1:T}|\mathbf{x}) = \prod_i q(\mathbf{z}_i|f_\phi^q(\mathbf{z}_{<i}, \mathbf{x}))q(\mathbf{z}_i^{cont})$$

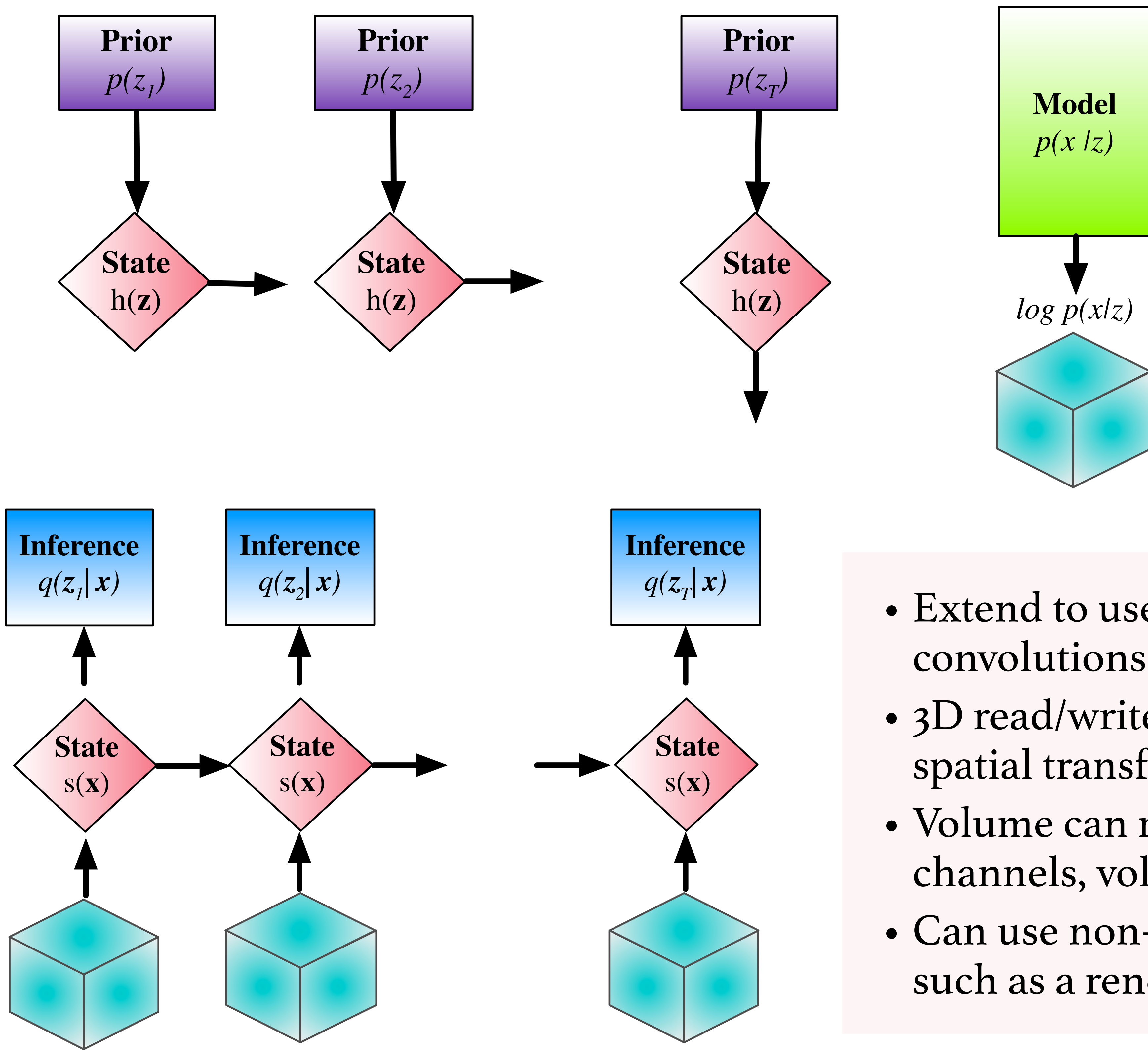
# Structured Sequential VAEs

*AIR - Attend-Infer-Repeat*



# Volumetric VAEs

*Volumetric DRAW*

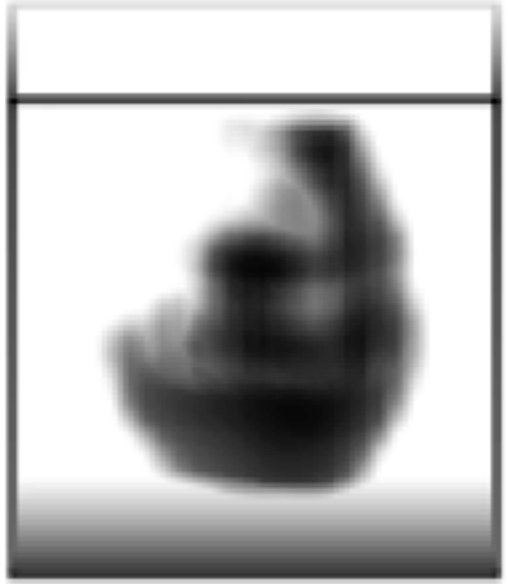


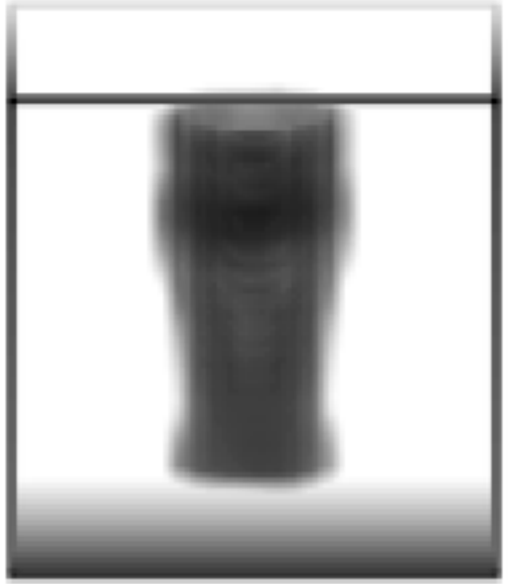

*Model can be non-differentiable, like a graphics engine.*

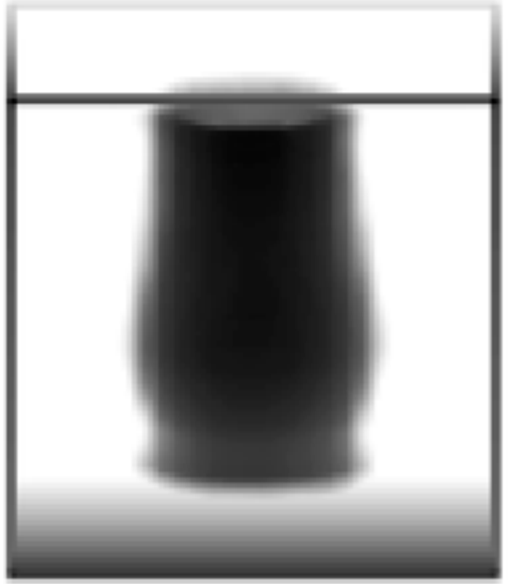
- Extend to use volumetric convolutions and canvas.
- 3D read/write attention using 3D spatial transformers.
- Volume can represent colour channels, volumes, time.
- Can use non-differentiable model such as a renderer.

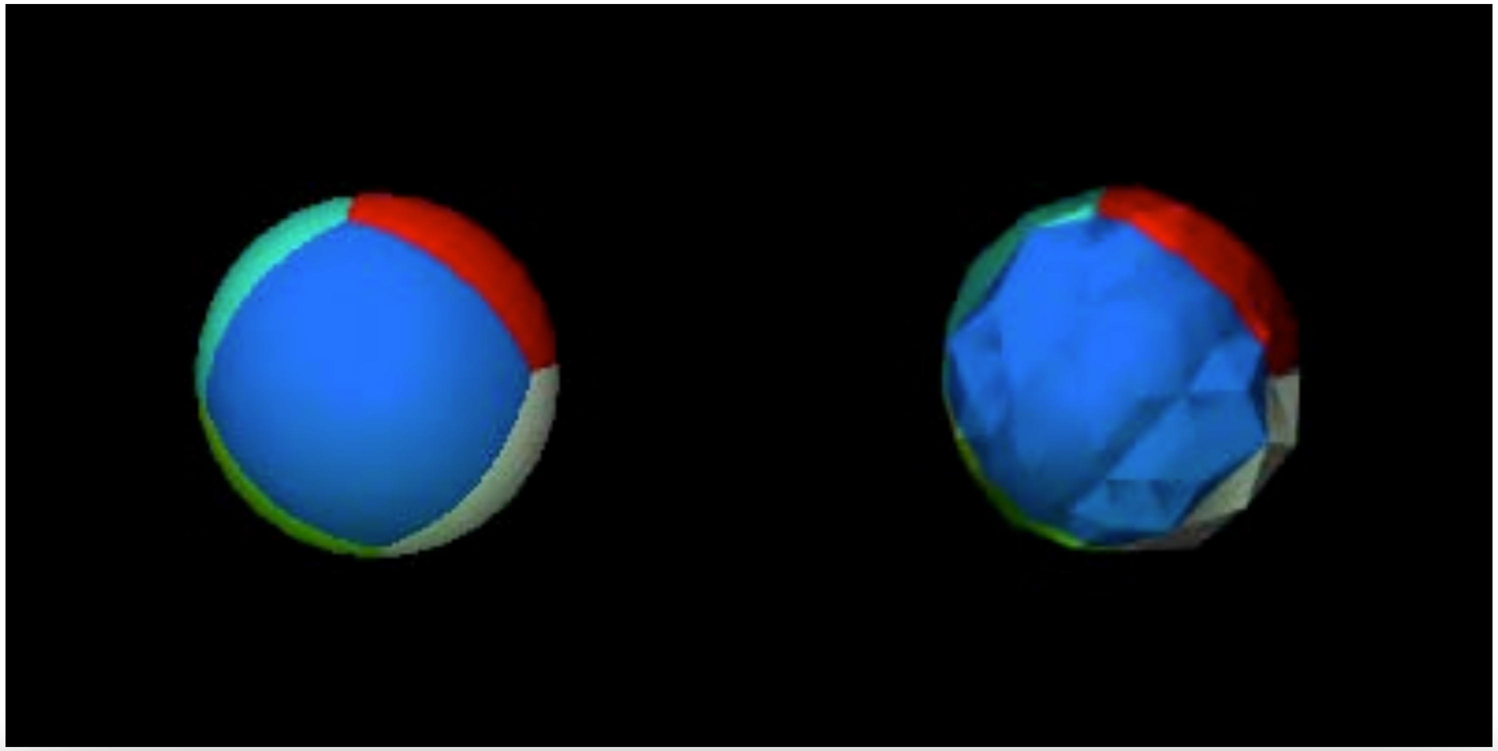
# Volumetric VAEs

Volumetric DRAW

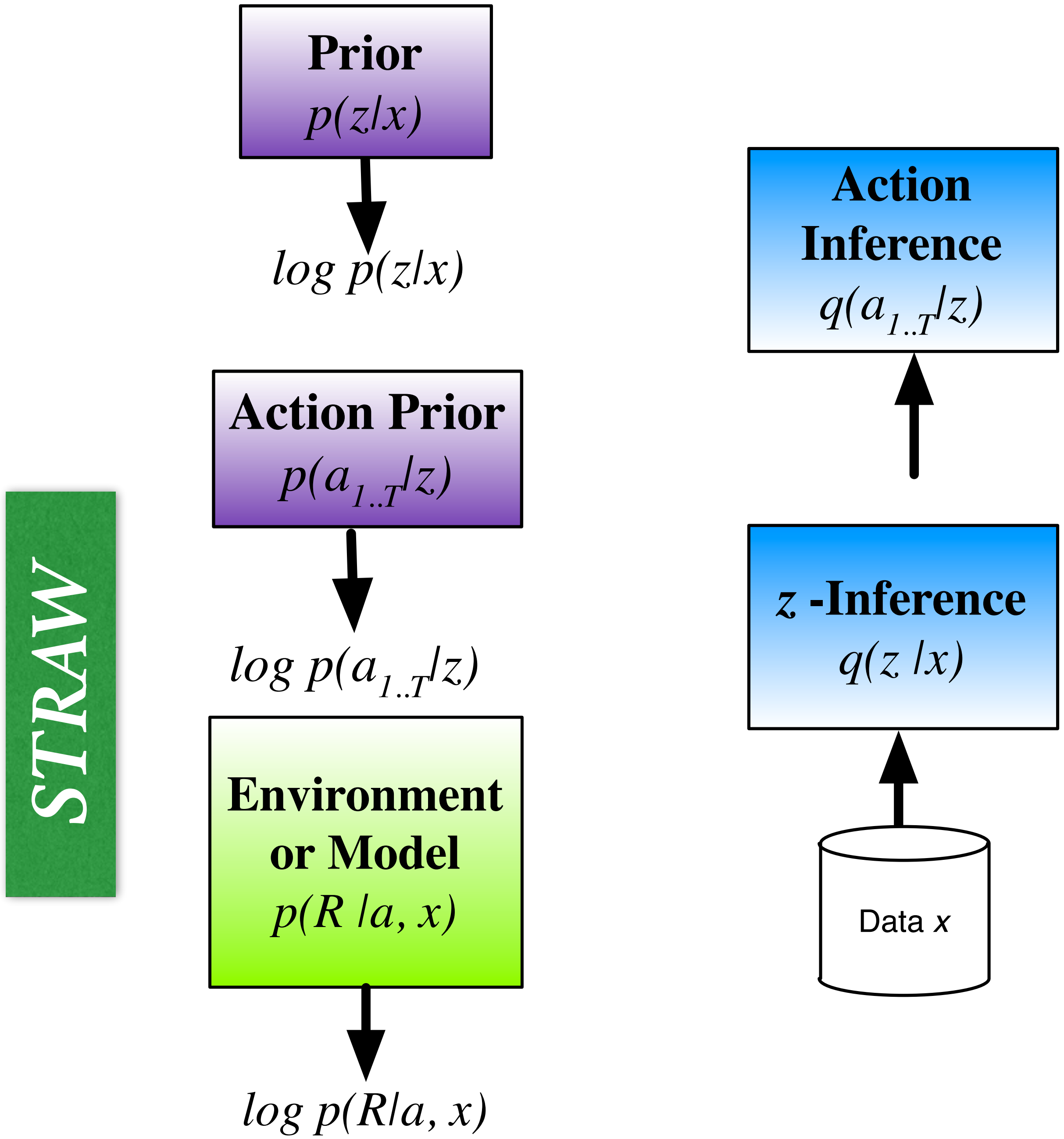








# Macro-action Learning




$$p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I})$$
$$p(\mathbf{a}_{1\dots T}|\mathbf{z}) = \mathcal{U}_n(a)$$

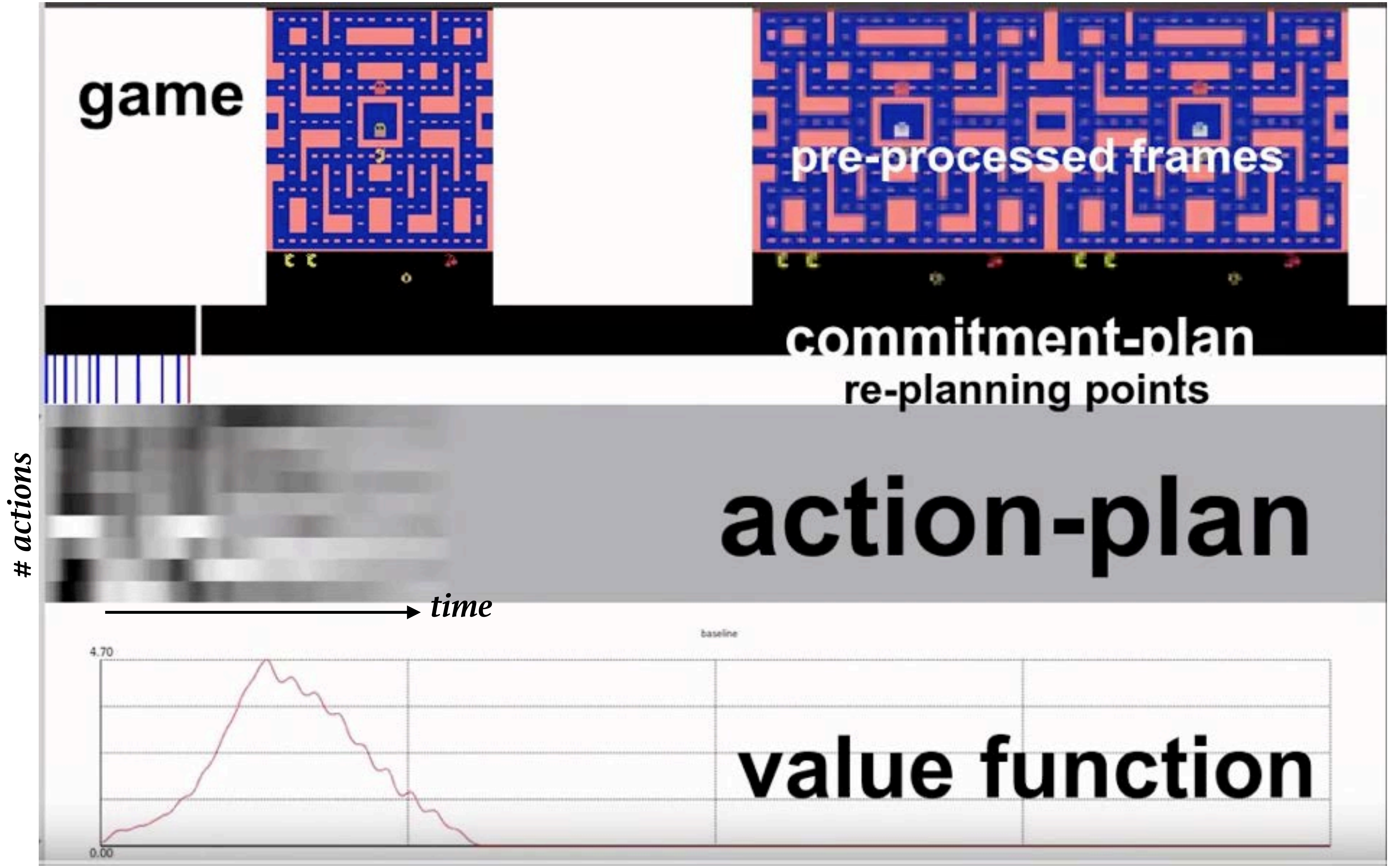
$$p(R|\mathbf{a}_{1\dots T}) \propto e^{\nu R(a,\mathbf{x})}$$

$$q(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}|\mu_\phi(\mathbf{x}), \Sigma_\phi(\mathbf{x}))$$
$$q(\mathbf{a}|\mathbf{z}) = Cat(\mathbf{a}|\pi_\theta(\mathbf{z}))$$

***Instance of a variational MDP***

$$\mathcal{F}^\pi(\theta) = \mathbb{E}_{q(a,z|x)}[R(a|x)] - \alpha KL[q_\theta(\mathbf{z}|\mathbf{x})\|p(\mathbf{z}|\mathbf{x})] + \alpha\mathbb{H}[\pi_\theta(\mathbf{a}|\mathbf{z})]$$

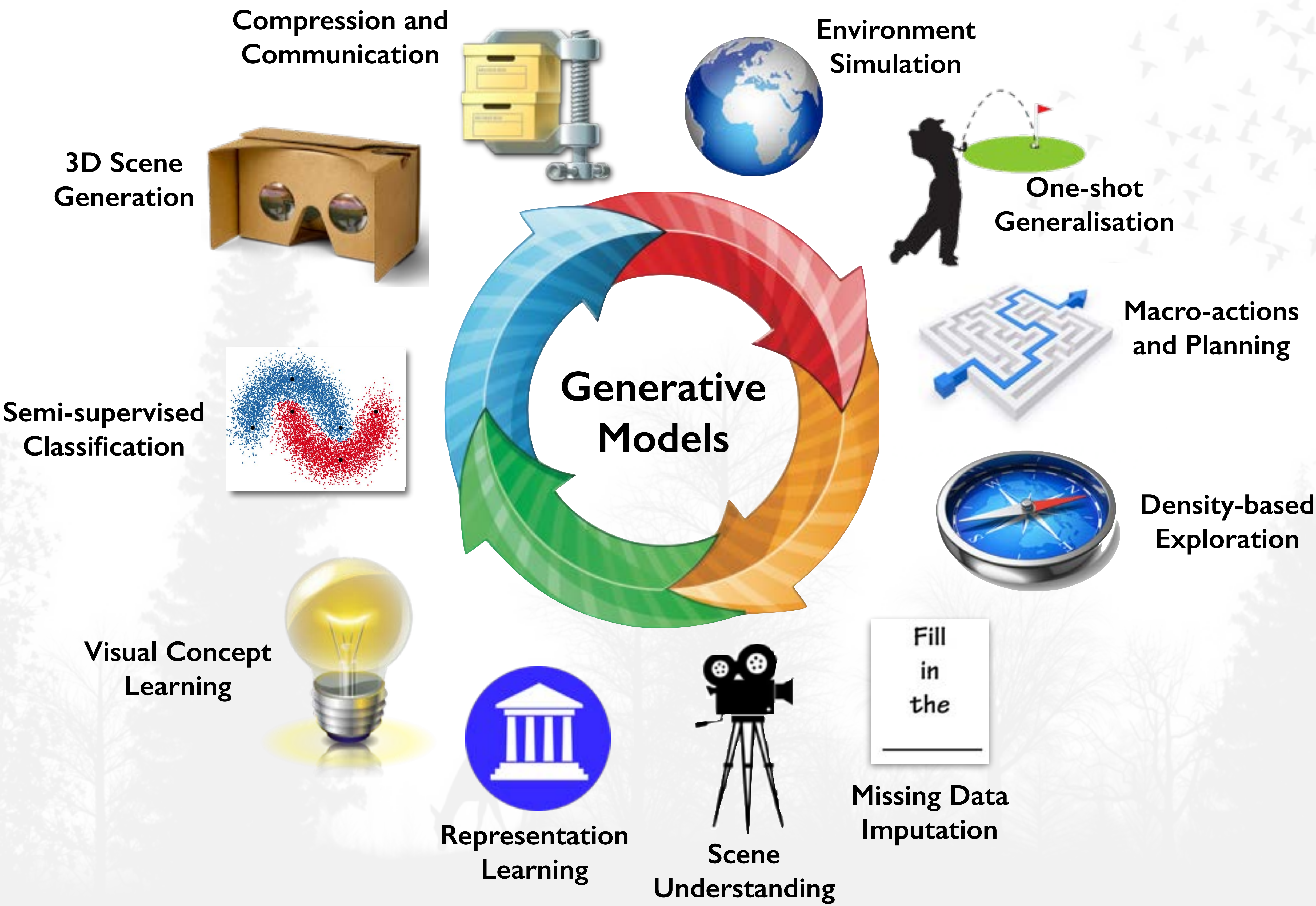
# Macro-action Learning



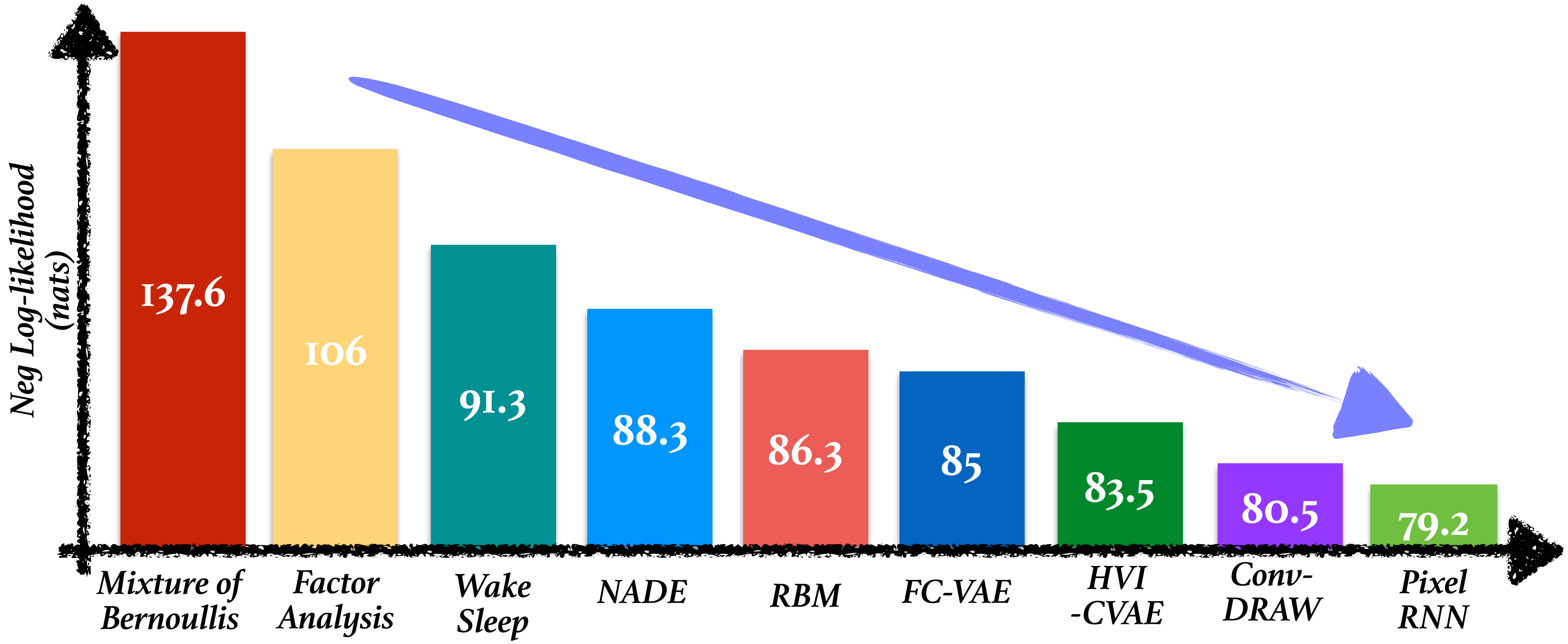
# Part VI

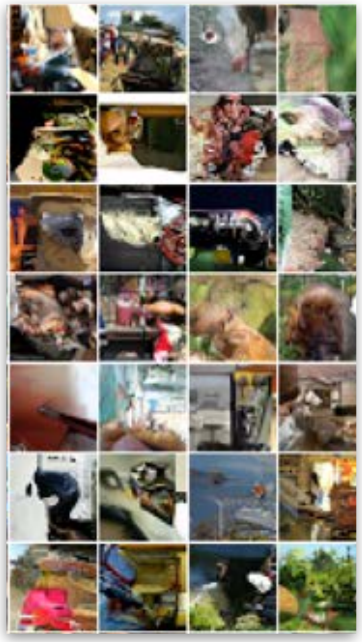
# Summary

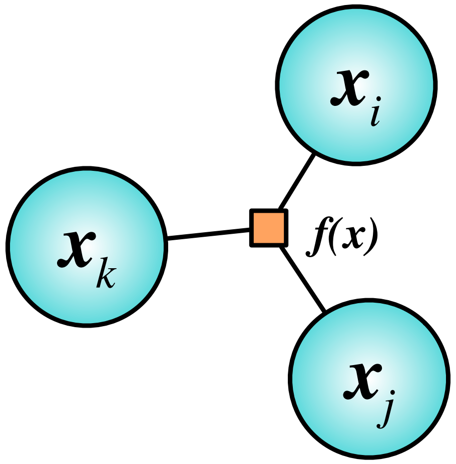
# Demonstrated Applications of Generative Models



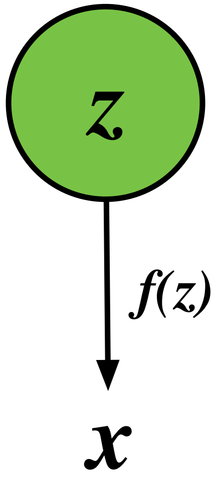
# Summary



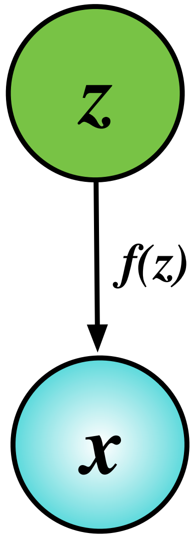






**Fully-observed models**

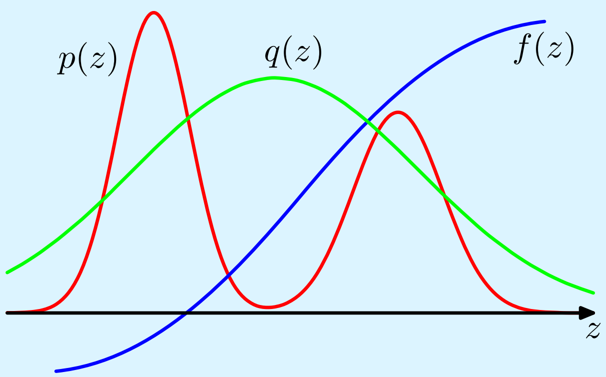



**Transformation models**



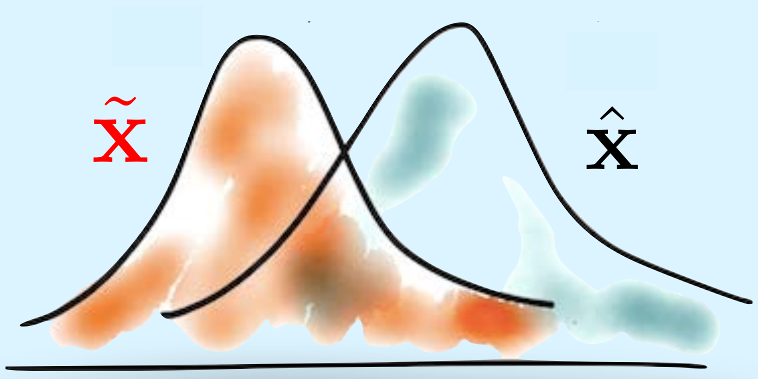

**Latent variable models**

# Summary




**Learning principle: Model Evidence**

$$p(\mathbf{x}) = \int p(\mathbf{x}, \mathbf{z}) d\mathbf{z}$$

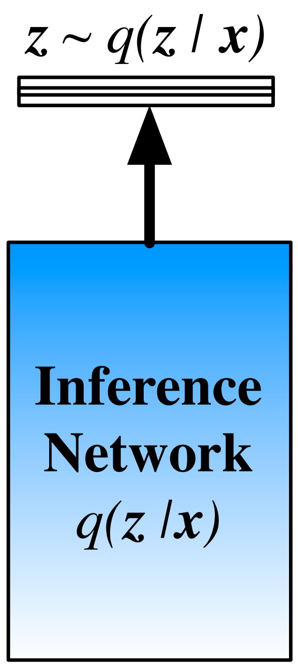



**Learning principle: Two-sample Tests**

$$\frac{p(\hat{\mathbf{x}})}{p(\tilde{\mathbf{x}})} = 1 \qquad p(\hat{\mathbf{x}}) = p(\tilde{\mathbf{x}})$$

# Summary

### *Amortised Inference*



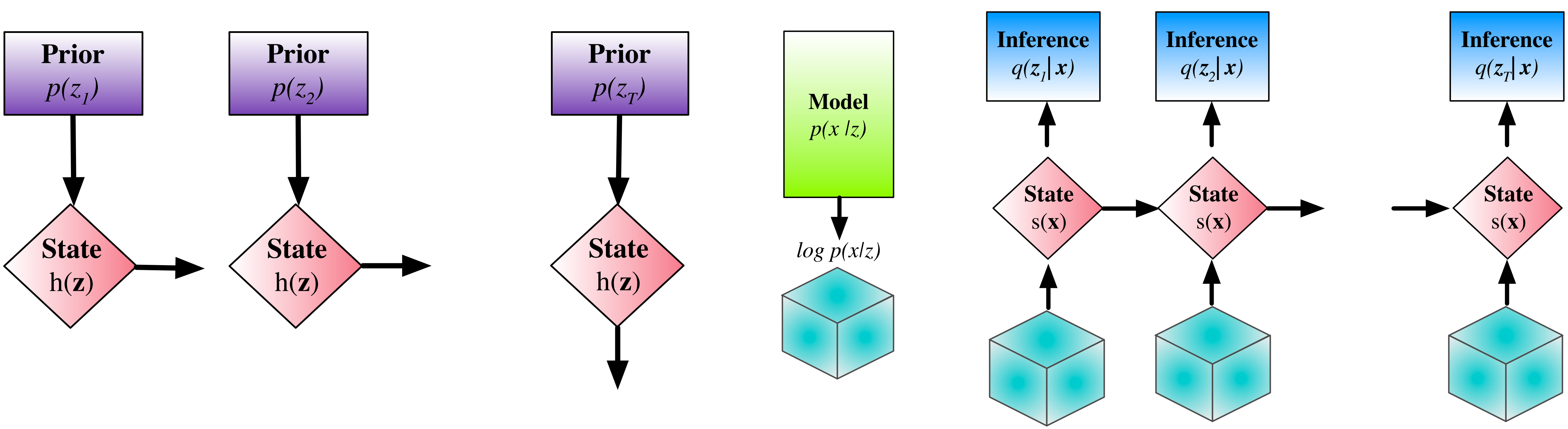

### *Stochastic optimisation*

$$\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{z})}[f_\theta(\mathbf{z})] = \nabla \int \boxed{q_\phi(\mathbf{z})}\boxed{f_\theta(\mathbf{z})} d\mathbf{z}$$

**Pathwise Estimator**
When easy to use transformation is available and differentiable function *f*.

**Score-function estimator**
When function *f* non-differentiable and *q(z)* is easy to sample from.

### *Families of VAEs*



# The Future of Generative Models

**In the aid of supervised and reward-based systems**
Calibration, confidence intervals, robustness and interpretability.

**Data-efficient learning systems**
Make more efficient use of scarce data

**Semi-parametric**
Combining parametric and non-parametric models for scalable, accurate, adaptive models

**Complementary systems and integrated agents**
Richer scene understanding
Self-directed and curious agents
Conceptual reasoning
Integrated planning and control systems

**Scientific discovery**
Exploratory analysis.
Synthesis and simulation: cosmic phenomena, climate systems.

# Building Machines that Imagine and Reason

## Principles and Applications of Deep Generative Models

Shakir Mohamed

**Thanks to many people:**
Danilo Rezende, Theophane Weber, Andriy Mnih, Ali Eslami, Karol Gregor, Sasha Veznevehts, Silvia Chiappa, Irina Higgins, Marc Bellemare, Charles Blundell, Benigno Uria, David Pfau, Lars Buesing, David Barret, Daan Wierstra, and many others at DeepMind.

Google DeepMind

joinus@deepmind.com

@shakir_za shakir@google.com

Deep Learning Summer School
August 2016

# Some References

Applications of Deep Generative Models


Rezende, Danilo Jimenez, Shakir Mohamed, and Daan Wierstra. "Stochastic backpropagation and approximate inference in deep generative models." ICML 2014

Kingma, Diederik P., and Max Welling. "Auto-encoding variational bayes." ICLR 2014

Gregor, Karol, et al. "Towards Conceptual Compression." arXiv preprint arXiv:1604.08772 (2016).

Eslami, S. M., Heess, N., Weber, T., Tassa, Y., Kavukcuoglu, K., & Hinton, G. E. (2016). Attend, Infer, Repeat: Fast Scene Understanding with Generative Models. arXiv preprint arXiv:1603.08575.

Oh, Junhyuk, Xiaoxiao Guo, Honglak Lee, Richard L. Lewis, and Satinder Singh. "Action-conditional video prediction using deep networks in atari games." In Advances in Neural Information Processing Systems, pp. 2863-2871. 2015.

Rezende, Danilo Jimenez, Shakir Mohamed, Ivo Danihelka, Karol Gregor, and Daan Wierstra. "One-Shot Generalization in Deep Generative Models." arXiv preprint arXiv:1603.05106 (2016).

Rezende, Danilo Jimenez, S. M. Eslami, Shakir Mohamed, Peter Battaglia, Max Jaderberg, and Nicolas Heess. "Unsupervised Learning of 3D Structure from Images." arXiv preprint arXiv:1607.00662 (2016).

Kingma, Diederik P., Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. "Semi-supervised learning with deep generative models." In Advances in Neural Information Processing Systems, pp. 3581-3589. 2014.

Maaløe, Lars, Casper Kaae Sønderby, Søren Kaae Sønderby, and Ole Winther. "Auxiliary Deep Generative Models." arXiv preprint arXiv:1602.05473 (2016).

Odena, Augustus. "Semi-Supervised Learning with Generative Adversarial Networks." arXiv preprint arXiv:1606.01583 (2016).

Springenberg, Jost Tobias. "Unsupervised and Semi-supervised Learning with Categorical Generative Adversarial Networks." arXiv preprint arXiv:1511.06390 (2015).

Blundell, Charles, Benigno Uria, Alexander Pritzel, Yazhe Li, Avraham Ruderman, Joel Z. Leibo, Jack Rae, Daan Wierstra, and Demis Hassabis. "Model-Free Episodic Control." arXiv preprint arXiv:1606.04460 (2016).

Higgins, Irina, Loic Matthey, Xavier Glorot, Arka Pal, Benigno Uria, Charles Blundell, Shakir Mohamed, and Alexander Lerchner. "Early Visual Concept Learning with Unsupervised Deep Learning." arXiv preprint arXiv:1606.05579 (2016).

Bellemare, Marc G., Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. "Unifying Count-Based Exploration and Intrinsic Motivation." arXiv preprint arXiv:1606.01868 (2016).

# Some References


Alexander (Sasha) Vezhnevets, Mnih, Volodymyr, John Agapiou, Simon Osindero, Alex Graves, Oriol Vinyals, and Koray Kavukcuoglu. "Strategic Attentive Writer for Learning Macro-Actions." arXiv preprint arXiv:1606.04695 (2016).

Gregor, Karol, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan Wierstra. "DRAW: A recurrent neural network for image generation." arXiv preprint arXiv:1502.04623 (2015).

### Fully-observed Models

Oord, Aaron van den, Nal Kalchbrenner, and Koray Kavukcuoglu. "Pixel recurrent neural networks." arXiv preprint arXiv:1601.06759 (2016).

Larochelle, Hugo, and Iain Murray. "The Neural Autoregressive Distribution Estimator." In AISTATS, vol. 1, p. 2. 2011.

Uria, Benigno, Iain Murray, and Hugo Larochelle. "A Deep and Tractable Density Estimator." In ICML, pp. 467-475. 2014.

Veness, Joel, Kee Siong Ng, Marcus Hutter, and Michael Bowling. "Context tree switching." In 2012 Data Compression Conference, pp. 327-336. IEEE, 2012.

Rue, Havard, and Leonhard Held. Gaussian Markov random fields: theory and applications. CRC Press, 2005.

Wainwright, Martin J., and Michael I. Jordan. "Graphical models, exponential families, and variational inference." Foundations and Trends® in Machine Learning 1, no. 1-2 (2008): 1-305.

### Transformation Models

Tabak, E. G., and Cristina V. Turner. "A family of nonparametric density estimation algorithms." Communications on Pure and Applied Mathematics 66, no. 2 (2013): 145-164.

Rezende, Danilo Jimenez, and Shakir Mohamed. "Variational inference with normalizing flows." arXiv preprint arXiv:1505.05770 (2015).

Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial nets." In Advances in Neural Information Processing Systems, pp. 2672-2680. 2014.

Verrelst, Herman, Johan Suykens, Joos Vandewalle, and Bart De Moor. "Bayesian learning and the Fokker-Planck machine." In Proceedings of the International Workshop on Advanced Black-box Techniques for Nonlinear Modeling, Leuven, Belgium, pp. 55-61. 1998.

Devroye, Luc. "Random variate generation in one line of code." In Proceedings of the 28th conference on Winter simulation, pp. 265-272. IEEE Computer Society, 1996.

# Some References

## Latent variable models


Dayan, Peter, Geoffrey E. Hinton, Radford M. Neal, and Richard S. Zemel. "The helmholtz machine." Neural computation 7, no. 5 (1995): 889-904.

Hyvärinen, A., Karhunen, J., & Oja, E. (2004). Independent component analysis (Vol. 46). John Wiley & Sons.

Gregor, Karol, Ivo Danihelka, Andriy Mnih, Charles Blundell, and Daan Wierstra. "Deep autoregressive networks." arXiv preprint arXiv:1310.8499 (2013).

Ghahramani, Zoubin, and Thomas L. Griffiths. "Infinite latent feature models and the Indian buffet process." In Advances in neural information processing systems, pp. 475-482. 2005.

Teh, Yee Whye, Michael I. Jordan, Matthew J. Beal, and David M. Blei. "Hierarchical dirichlet processes." Journal of the american statistical association (2012).

Adams, Ryan Prescott, Hanna M. Wallach, and Zoubin Ghahramani. "Learning the Structure of Deep Sparse Graphical Models." In AISTATS, pp. 1-8. 2010.

Lawrence, Neil D. "Gaussian process latent variable models for visualisation of high dimensional data." Advances in neural information processing systems 16.3 (2004): 329-336.

Damianou, Andreas C., and Neil D. Lawrence. "Deep Gaussian Processes." In AISTATS, pp. 207-215. 2013.

Mattos, César Lincoln C., Zhenwen Dai, Andreas Damianou, Jeremy Forth, Guilherme A. Barreto, and Neil D. Lawrence. "Recurrent Gaussian Processes." arXiv preprint arXiv:1511.06644 (2015).

Salakhutdinov, Ruslan, Andriy Mnih, and Geoffrey Hinton. "Restricted Boltzmann machines for collaborative filtering." In Proceedings of the 24th international conference on Machine learning, pp. 791-798. ACM, 2007.

Saul, Lawrence K., Tommi Jaakkola, and Michael I. Jordan. "Mean field theory for sigmoid belief networks." Journal of artificial intelligence research 4, no. 1 (1996): 61-76.

Frey, Brendan J., and Geoffrey E. Hinton. "Variational learning in nonlinear Gaussian belief networks." Neural Computation 11, no. 1 (1999): 193-213.

# Some References

### Inference and Learning


Jordan, Michael I., Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. "An introduction to variational methods for graphical models." Machine learning 37, no. 2 (1999): 183-233.

Hoffman, Matthew D., David M. Blei, Chong Wang, and John William Paisley. "Stochastic variational inference." Journal of Machine Learning Research 14, no. 1 (2013): 1303-1347.

Honkela, Antti, and Harri Valpola. "Variational learning and bits-back coding: an information-theoretic view to Bayesian learning." IEEE Transactions on Neural Networks 15, no. 4 (2004): 800-810.

Burda, Yuri, Roger Grosse, and Ruslan Salakhutdinov. "Importance weighted autoencoders." arXiv preprint arXiv:1509.00519 (2015).

Li, Yingzhen, and Richard E. Turner. "Variational Inference with R\'enyi Divergence." arXiv preprint arXiv:1602.02311 (2016).

Borgwardt, Karsten M., and Zoubin Ghahramani. "Bayesian two-sample tests." arXiv preprint arXiv:0906.4032 (2009).

Gutmann, Michael, and Aapo Hyvärinen. "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models." AISTATS. Vol. 1. No. 2. 2010.

Tsuboi, Yuta, Hisashi Kashima, Shohei Hido, Steffen Bickel, and Masashi Sugiyama. "Direct Density Ratio Estimation for Large-scale Covariate Shift Adaptation." Information and Media Technologies 4, no. 2 (2009): 529-546.

Sugiyama, Masashi, Taiji Suzuki, and Takafumi Kanamori. Density ratio estimation in machine learning. Cambridge University Press, 2012.


### Amortised Inference


Gershman, Samuel J., and Noah D. Goodman. "Amortized inference in probabilistic reasoning." In Proceedings of the 36th Annual Conference of the Cognitive Science Society. 2014.

Rezende, Danilo Jimenez, Shakir Mohamed, and Daan Wierstra. "Stochastic backpropagation and approximate inference in deep generative models." arXiv preprint arXiv:1401.4082 (2014).

Heess, Nicolas, Daniel Tarlow, and John Winn. "Learning to pass expectation propagation messages." In Advances in Neural Information Processing Systems, pp. 3219-3227. 2013.

Jitkrittum, Wittawat, Arthur Gretton, Nicolas Heess, S. M. Eslami, Balaji Lakshminarayanan, Dino Sejdinovic, and Zoltán Szabó. "Kernel-based just-in-time learning for passing expectation propagation messages." arXiv preprint arXiv:1503.02551 (2015).

Korattikara, Anoop, Vivek Rathod, Kevin Murphy, and Max Welling. "Bayesian dark knowledge." arXiv preprint arXiv:1506.04416 (2015).

# Some References

**Stochastic Optimisation**


P L'Ecuyer, Note: On the interchange of derivative and expectation for likelihood ratio derivative estimators, Management Science, 1995

Peter W Glynn, Likelihood ratio gradient estimation for stochastic systems, Communications of the ACM, 1990

Michael C Fu, Gradient estimation, Handbooks in operations research and management science, 2006

Ronald J Williams, Simple statistical gradient-following algorithms for connectionist reinforcement learning, Machine learning, 1992

Paul Glasserman, Monte Carlo methods in financial engineering, , 2003

Luc Devroye, Random variate generation in one line of code, Proceedings of the 28th conference on Winter simulation, 1996

L. Devroye, Non-uniform random variate generation, , 1986

Omiros Papaspiliopoulos, Gareth O Roberts, Martin Skold, A general framework for the parametrization of hierarchical models, Statistical Science, 2007

Michael C Fu, Gradient estimation, Handbooks in operations research and management science, 2006

Ranganath, Rajesh, Sean Gerrish, and David M. Blei. "Black Box Variational Inference." In AISTATS, pp. 814-822. 2014.

Mnih, Andriy, and Karol Gregor. "Neural variational inference and learning in belief networks." arXiv preprint arXiv:1402.0030 (2014).

Lázaro-Gredilla, Miguel. "Doubly stochastic variational Bayes for non-conjugate inference." (2014).

Wingate, David, and Theophane Weber. "Automated variational inference in probabilistic programming." arXiv preprint arXiv:1301.1299 (2013).

Paisley, John, David Blei, and Michael Jordan. "Variational Bayesian inference with stochastic search." arXiv preprint arXiv:1206.6430 (2012).