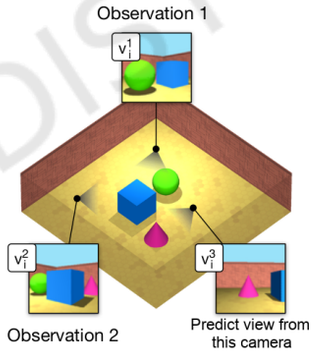
kptan86@gmail.com

DQN

"A computer vision system predicts how a 3D scene looks from any viewpoint after just a few 2D views from other viewpoints."

Objective: Predict the image view by a camera (with arbitrary position and angles), given some image samples taken at different positions and angles in the same scene.



"Neural scene representation and rendering". Ali Eslami et al, Science 2018

YouTube Video

https://www.youtube.com/watch?v=G-kWNQJ4idw&feature=youtu.be (https://www.youtube.com/watch?v=G-kWNQJ4idw&feature=youtu.be)

Assumption

- The positions (and other properties, e.g. colors) of the 3D objects can be represented as a high-dimensional Gaussian distribution
- can be thought as a varaint of variational autoencoder (but conditioned with camera position and angles)

(Personal) Comments

Pros

- Showcased advanced algorithm cocktailing
  - RNN
  - CNN
  - Latent Gaussian mixture model
  - Variational autoencoder
- Hint* (by luck?) on neural information combination

Cons

- Unncessarily complicated
- Theorectically unsound, difficult to understand / generalize
- Unsure of limit in practical uses

DO NOT DISTRIBUTE

kptan86@gmail.com

(more) Formal solution

Inference model

1. Recognize that the *ground truth context* to model is simply **3D objects in a room**
2. Use a representation z (e.g. Gaussian, polygons, grids etc) to model the 3D objects
3. Find the most probable values (or distribution) for the parameters of the presentation
   - e.g. minimize reconstruction loss with gradient descent or other optimzation technique
   - (something along the line of $z = \text{argmin}_z \sum_i \text{Loss}(\text{Proj}(G(z)) - X_i) + \lambda(G(z))$

Generative model

1. Simply use the reconstruction method on the learned representation at the specified viewpoint v_s
   - (something along the line of $y = \text{Proj}(G(z))$)

3D projection

- 3D projection to 2D image is a well-studied problem
- The pixel in rendered 2D plane ($\mathbf{b}$) relates with a point in 3D ($\mathbf{x}, \mathbf{y}, \mathbf{z}$) (origin set at the camera position) with camera angles ($\theta_x, \theta_y, \theta_z$) by

$$\mathbf{b}_x = \mathbf{e_z}(\mathbf{d_x}/\mathbf{d_y}) + \mathbf{e_x}$$
$$\mathbf{b}_y = \mathbf{e_z}(\mathbf{d_y}/\mathbf{d_z}) + \mathbf{e_y}$$

$$\mathbf{d}_x = \cos\theta_y(\sin\theta_z\mathbf{y} + \cos\theta_z\mathbf{x}) - \sin\theta_y\mathbf{z}$$
$$\mathbf{d}_y = \sin\theta_x(\cos\theta_y\mathbf{z} + \sin\theta_y(\sin\theta_z\mathbf{y} + \cos\theta_z\mathbf{x})) + \cos\theta_x(\cos\theta_z\mathbf{y} - \sin\theta_z\mathbf{x})$$
$$\mathbf{d}_z = \cos\theta_x(\cos\theta_y\mathbf{z} + \sin\theta_y(\sin\theta_z\mathbf{y} + \cos\theta_z\mathbf{x})) - \sin\theta_x(\cos\theta_z\mathbf{y} - \sin\theta_z\mathbf{x})$$

where (e_x, e_y, e_z) are some constants for a fixed display setup

- **Given the basic trigonometric representation of θs, the projection is a (simply) a linear transformation**.

DO NOT DISTRIBUTE

kptan86@gmail.com

Neural network solution

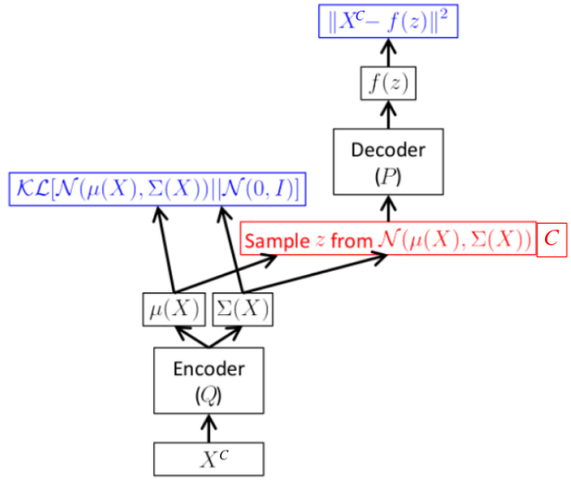
Inference model

1. Recognize that the *ground truth context* to model is simply **3D objects in a room**
2. Use a representation $G(z)$ (e.g. Gaussian, polygons, grids etc) to model the 3D objects
3. Find the most probable values (or distribution) for the parameters of the presentation
   - e.g. minimize reconstruction loss with gradient descent or other optimzation technique
   - (something along the line of $z = \text{argmin}_z \sum_i \text{Loss}(\text{Proj}(G(z)) - X_i) + \lambda(G(z))$
4. **Replace $G(x)$ with a (encoding) neural network**

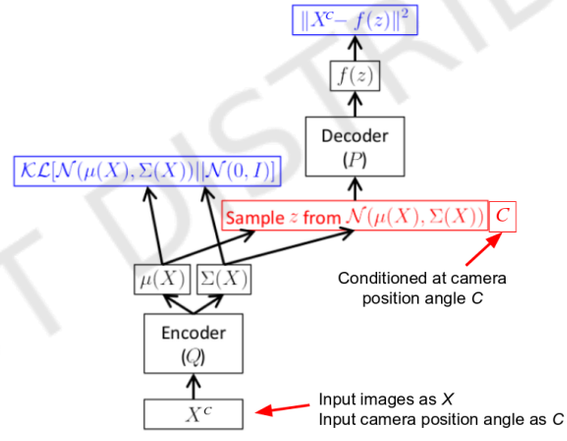
Generative model

1. Simply use the reconstruction method on the learned representation at the specified viewpoint v_s
   - (something along the line of $y = \text{Proj}(G(z))$)
2. **Replace $\text{Proj}(x)$ with a (generative) neural network**

From CVAE to DQN



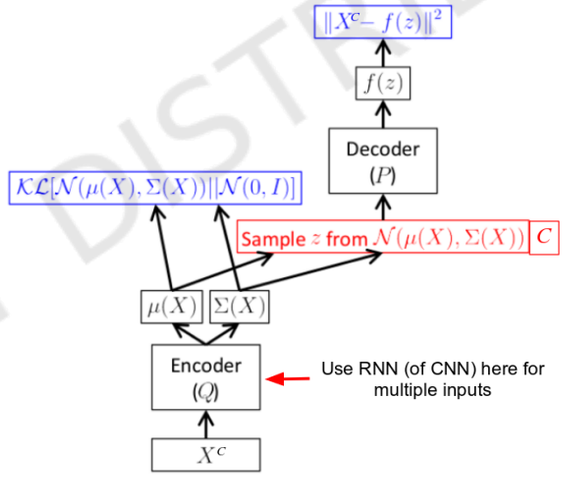
From CVAE to DQN



kptan86@gmail.com

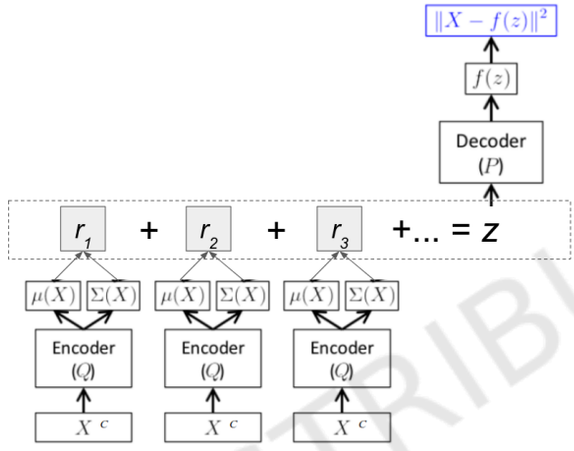
From CVAE to DQN

- The encoder could be enhanced to take variable number of inputs
  - use RNN (LTSM is used in DQN)
  - for $G(z)$ Recurrent latent Gaussian is used



Interestingly

- "**The additive aggregation function was found to work well in practice, despite its simplicity.**"
  - (A.K.A. "For some reason it works")
- (still a mystery to me)
  - (partially resolved by thinking of z as an array of binary encoding key)



DO NOT DISTRIBUTE