Conditional Generative Flow

Zhiyuan Tang

CSLT Report 20191217

How to use condition

- Joint probability density
- Condition in (change the net directly)
- Condition out (change the net via BP)
- Some typical types

Class-conditional (simple label to f)

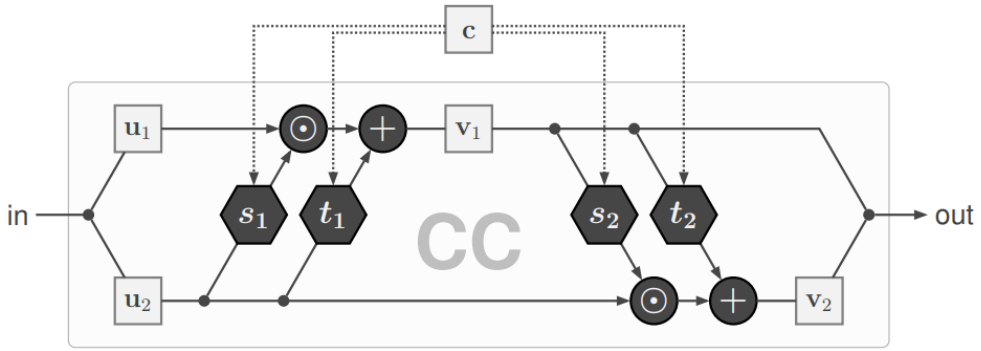


Figure 2: One conditional affine coupling block (CC).

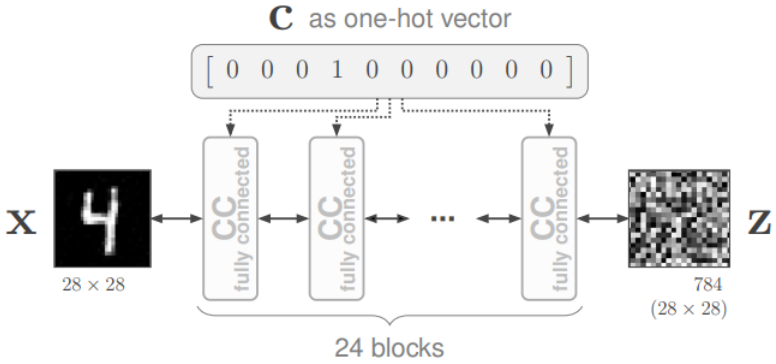


Figure 5: cINN model for conditional MNIST generation.

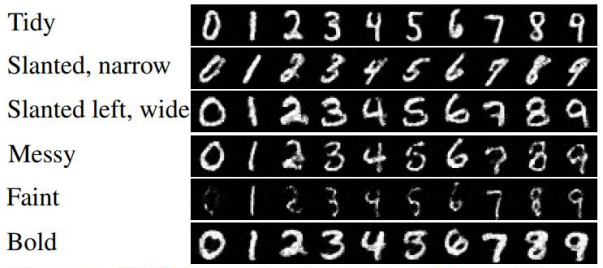


Figure 6: MNIST samples from our cINN conditioned on digit labels. All ten digits within one row $(0, \ldots, 9)$ were generated using the same latent code $\mathbf{z}$, but changing condition $\mathbf{c}$. We see that each $\mathbf{z}$ encodes a single style consistently across digits, while varying $\mathbf{z}$ between rows leads to strong differences in writing style.

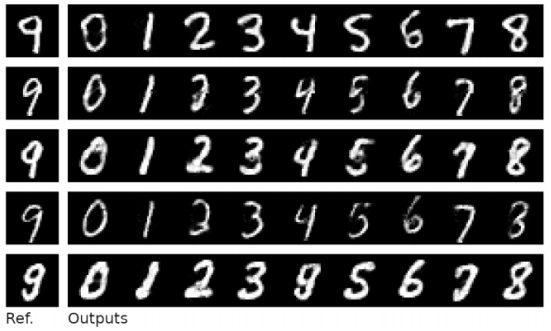


Figure 7: To perform style transfer, we determine the latent code $\mathbf{z} = f(\mathbf{x}; \mathbf{c}, \theta)$ of a validation image *(left)*, then use the inverse network $g = f^{-1}$ with different conditions $\hat{\mathbf{c}}$ to generate the other digits in the same style, $\hat{\mathbf{x}} = g(\mathbf{z}; \hat{\mathbf{c}}, \theta)$.

Guided image generation with **conditional** invertible neural networks

Class-conditional (to f, via pretrained model)

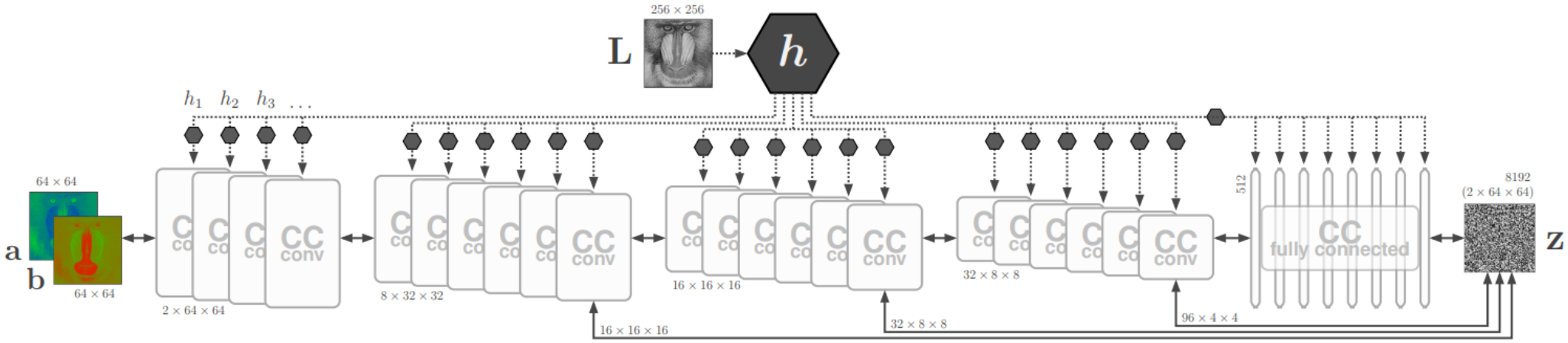
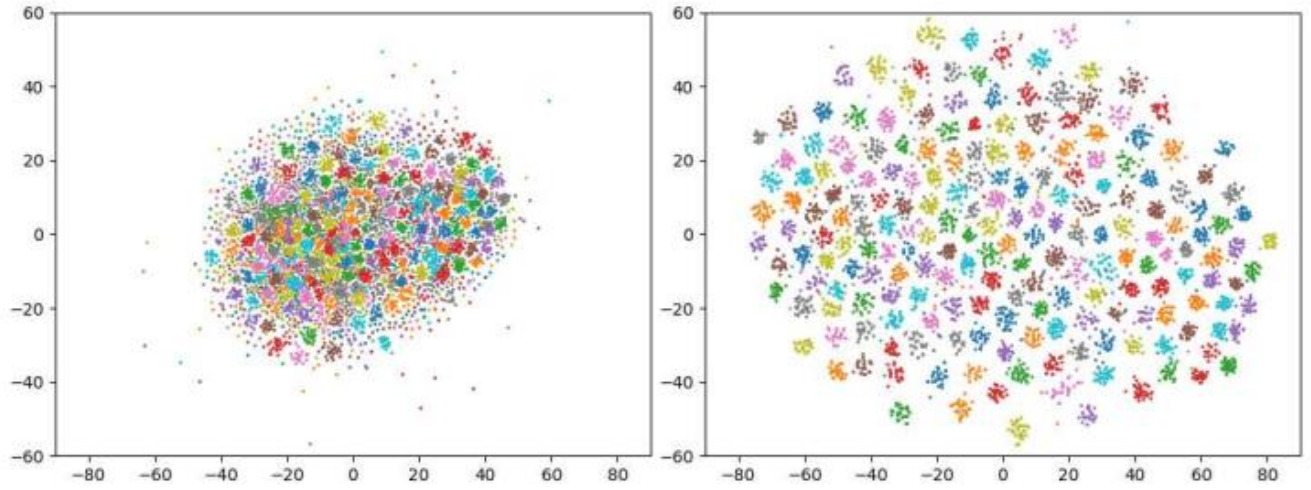


Figure 8: cINN model for diverse colorization. The conditioning network h consists of a truncated VGG [37] pretrained to predict colors on ImageNet, with separate convolutional heads $h_1, h_2, h_3, \ldots$ tailoring the extracted features to each individual conditional coupling block (CC). After each group of coupling blocks, we apply Haar wavelet downsampling (Fig. 3) to reduce the spatial dimensions and, where indicated by arrows, split off parts of the latent code $\mathbf{z}$ early.

VGG-like (discriminative) *h* is pretrained to classify each pixel of the gray image into color bins, then fixed.

Guided image generation with **conditional** invertible neural networks

Conditional adversarial generative flow (to z)



(a) Inferred (b) Sampled

Figure 1. Barnes-Hut *t*-SNE [40] visualization of 6, 000 latent vectors on 200 identities of CGlow [17]. (a) latent vectors inferred by forward CGlow; (b) randomly sampled latent vectors by inverse CGlow. Best viewed in color.

The clusters of sampled latent vectors keep far apart and have a large divergence from the real distribution. The underlying distribution of image conditions is difficult to measure precisely on the latent space.

Conditional Adversarial Generative Flow for Controllable Image Synthesis, CVPR

Conditional adversarial generative flow (to z)

$$\log p(x, c_s) = \log p(z, c_s) + \log \left| \det \frac{dF}{dx} \right|, \qquad (4)$$

where we let c_s denote the conditions under supervision.

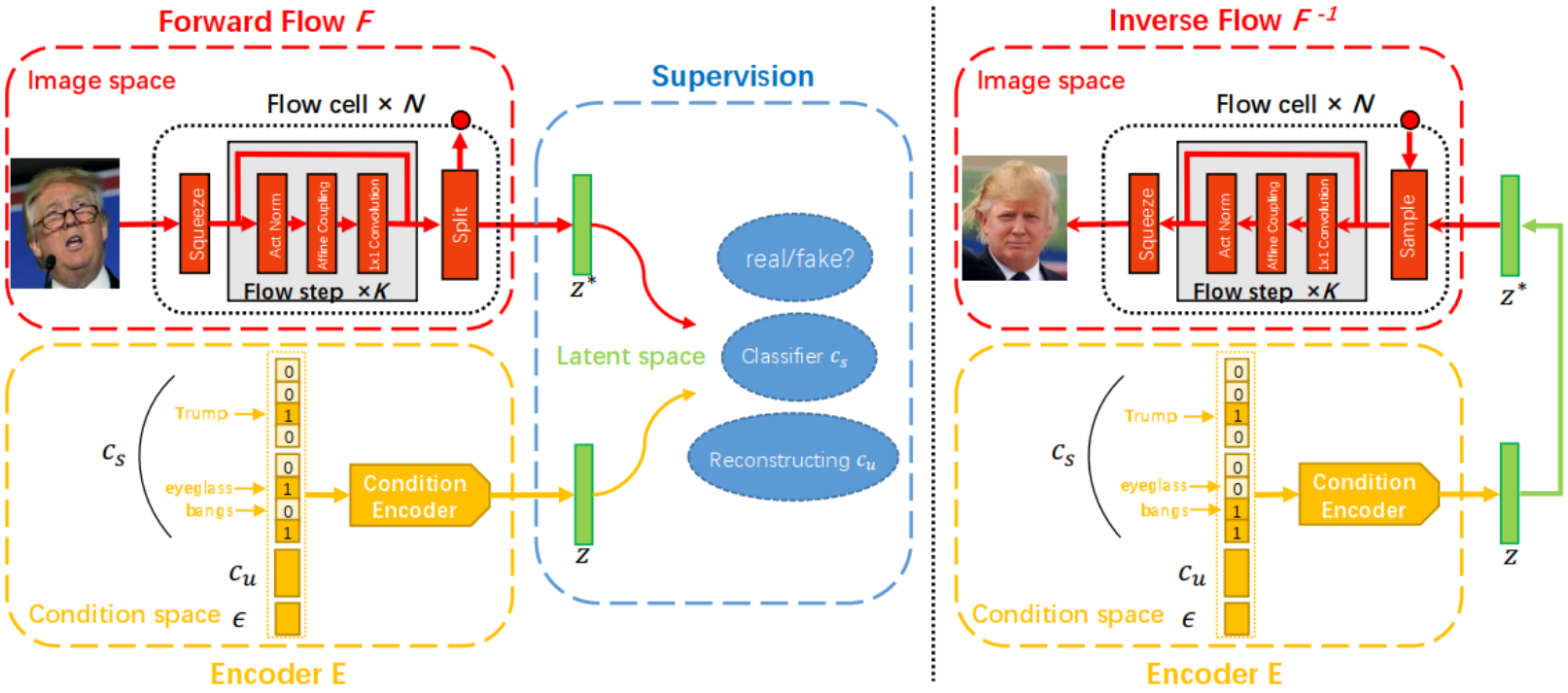
Using Bayesian formula, maximizing equation 4 is equal to:

$$\begin{aligned} \max \quad & \mathbb{E}_{z \sim p^*(z), c_s \sim p(c_s)} [\log p(c_s|z)] \\ & + \mathbb{E}_{z \sim p^*(z)} [\log p^*(z) + \log \left| \det \frac{dF}{dx} \right|], \end{aligned} \qquad (5)$$

where the prior $p^*(z)$ is modeled by a standard Gaussian distribution.

Conditional Adversarial Generative Flow for Controllable Image Synthesis, CVPR

Conditional adversarial generative flow (to z)



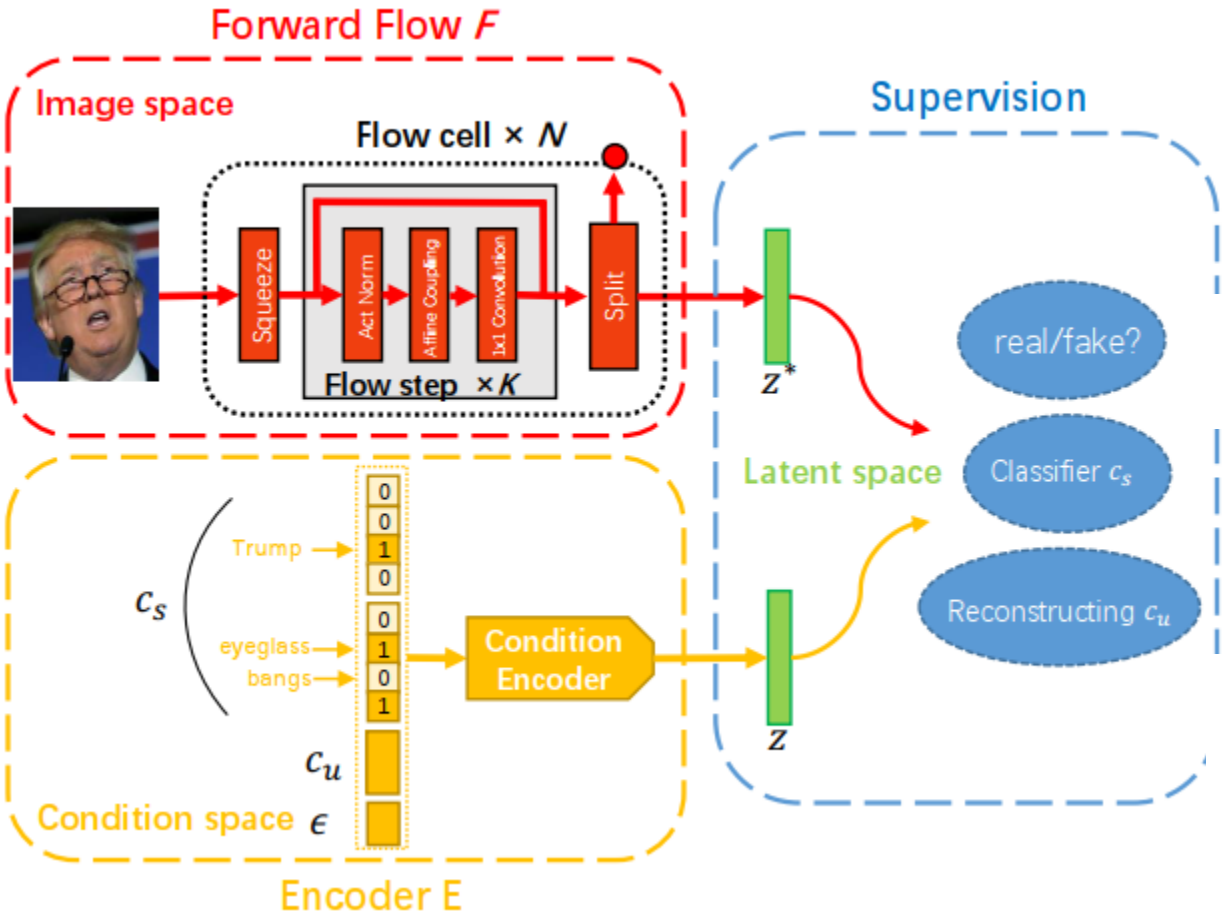
(a) The training procedure of CAGlow (b) The image synthesis process of CAGlow

Figure 2. Illustration of the network architecture of the proposed conditional adversarial generative flow. It contains a reversible flow F, an encoder E, and a supervision block including a discriminator D_i distinguishing real vectors from fake ones, a classifier C classifying supervised conditions correctly and a decoder D_e reconstructing unsupervised conditions.

How to choose a better z.

Conditional Adversarial Generative Flow for Controllable Image Synthesis, CVPR

Conditional adversarial generative flow (to z)



$$\mathcal{L}_F = -\mathbb{E}_{z\sim p^*(z)}[\log p^*(z) + \log \left| \det \frac{dF}{dx} \right|]. \quad (9)$$

$$\mathcal{L}_E = -\mathbb{E}_{\epsilon\sim p(\epsilon),\tilde{c}\sim p(\tilde{c})}[\log D_{i\phi}(E_\theta(\tilde{c},\epsilon))]. \quad (10)$$

$$\mathcal{L}_{D_i} = -\mathbb{E}_{z\sim p^*(z)}[\log D_{i\phi}(z)] - \mathbb{E}_{z\sim p_\theta(z)}[1 - \log D_{i\phi}(z)]. \quad (11)$$

$$\begin{aligned}\mathcal{L}_C = &-\mathbb{E}_{z\sim p^*(z),c_s\sim p(c_s)}[\log q_\phi(c_s|z)] \\ &-\mathbb{E}_{z\sim p_\theta(z),c_s\sim p(c_s)}[\log q_\phi(c_s|z)].\end{aligned} \quad (12)$$

$$\mathcal{L}_{D_e} = -\mathbb{E}_{z\sim p_\theta(z),c_u\sim p(c_u)}[\log q_\phi(c_u|z)], \quad (13)$$

Conditional Adversarial Generative Flow for Controllable Image Synthesis, CVPR

Conditional continuous normalizing (condition out)

$$\mathcal{J} = \mathcal{L}_{NLL}(\mathbf{x}|y) + \beta \mathcal{L}_{Xent}(\hat{y}, y), \tag{4}$$

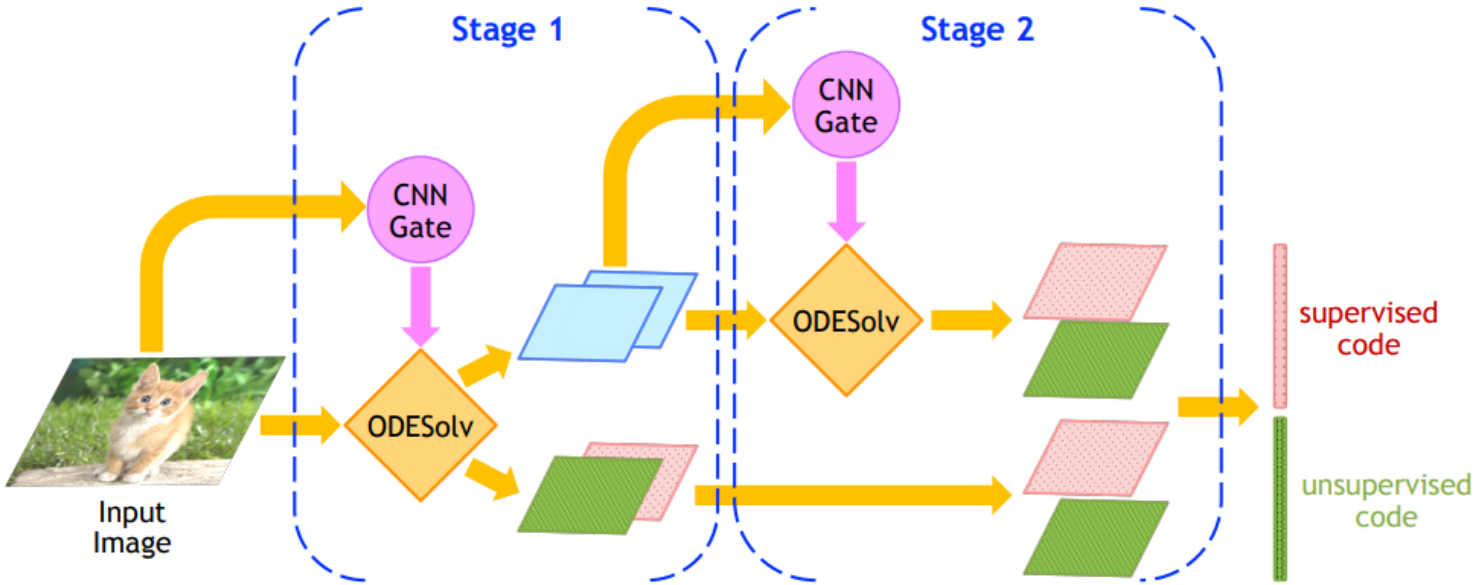


Figure 1: InfoCNF with CNN gates that learn the tolerances of the ODE solvers using reinforcement learning. The latent code in InfoCNF is split into the supervised and unsupervised codes. The supervised code is used for conditioning and classification. The unsupervised code captures other latent variations in the data.

InfoCNF: An efficient **conditional** continuous normalizing flow with adaptive solvers

Semi-Conditional (part condition)

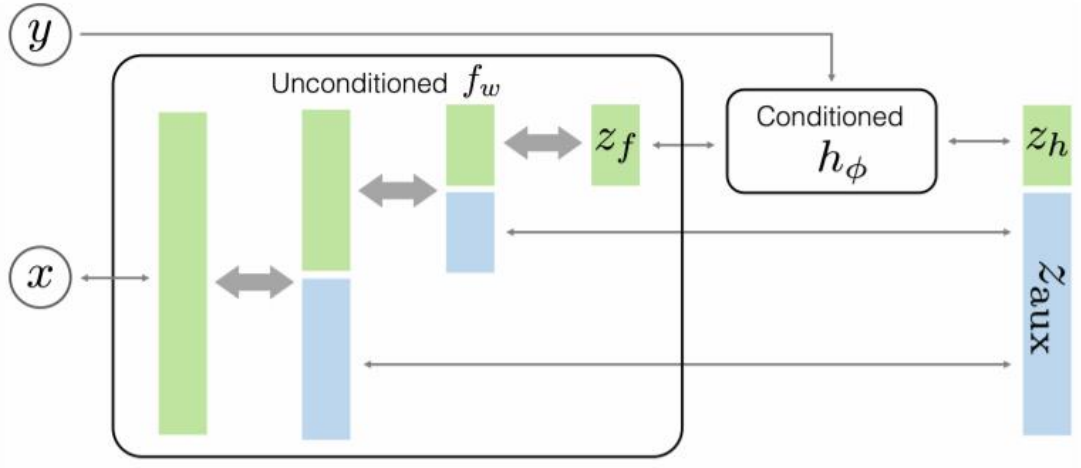


Figure 1. The proposed semi-conditional architecture consists of two parts: a large unconditional flow $f_w(x)$, and a relatively small conditional flow $h_\phi(z_f; y)$. The unconditional flow $f_w(x)$ is based on a multi-scale architecture and maps an input x into a low-dimensional z_f and an auxiliary vector z_{aux}. The conditional flow $h_\phi(z_f; y)$ maps the low-dimensional vector z_f to $z_h = h_\phi(z_f; y)$. The architecture allows to compute $p_\theta(x) = \mathbb{E}_y p_\theta(x, y)$ with a single forward pass of the computationally expensive flow f_θ and one pass of the inexpensive flow h_ϕ for every class label y.

$$\log p_\theta(x) = \log \left| \frac{\partial f_w(x)}{\partial x^T} \right| + \log \mathcal{N}(z_{\text{aux}}|0, I) \qquad (3)$$

$$+ \sum_{y=1}^{K} \log \left| \frac{\partial h_\phi(z_f; y)}{\partial z_f^T} \right| + \log \mathcal{N}(z_h|0, I) + \log p(y)$$

Semi-**Conditional** Normalizing Flows for Semi-Supervised Learning

Conditional Normalizing Flows (to z & f)

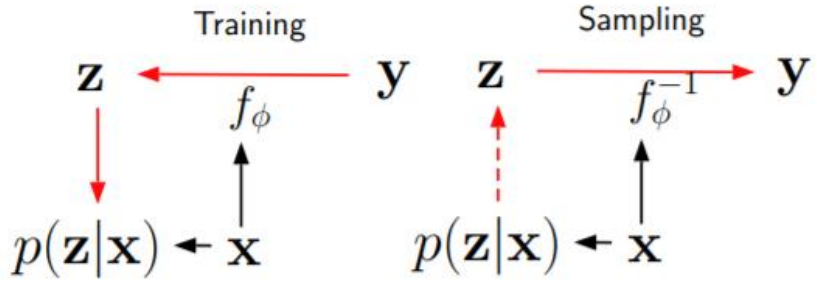


Figure 1: *Diagram of our model in the train and sampling phases. Solid lines represent deterministic mappings and dashed lines represent sampling. The conditioning variable enters the network in base density $p(\mathbf{z}|\mathbf{x})$ and the bijective mappings $f(\mathbf{y}, \mathbf{x})$.*

Conditional Prior	$p(\mathbf{z}\|\mathbf{x}) = \mathcal{N}(\mathbf{z}; \mu(\mathbf{x}), \sigma^2(\mathbf{x}))$
Conditional Split Prior	$p(\mathbf{z}_1\|\mathbf{z}_0, \mathbf{x}) = \mathcal{N}(\mathbf{z}_1; \mu(\mathbf{z}_0, \mathbf{x}), \sigma^2(\mathbf{z}_0, \mathbf{x}))$
Conditional Coupling	$\mathbf{y}_0 = s(\mathbf{z}_1, \mathbf{x}) \cdot \mathbf{z}_0 + t(\mathbf{z}_1, \mathbf{x}); \quad \mathbf{y}_1 = \mathbf{z}_1$

Learning Likelihoods with **Conditional** Normalizing Flows

Conditional recurrent flow (recurrent)

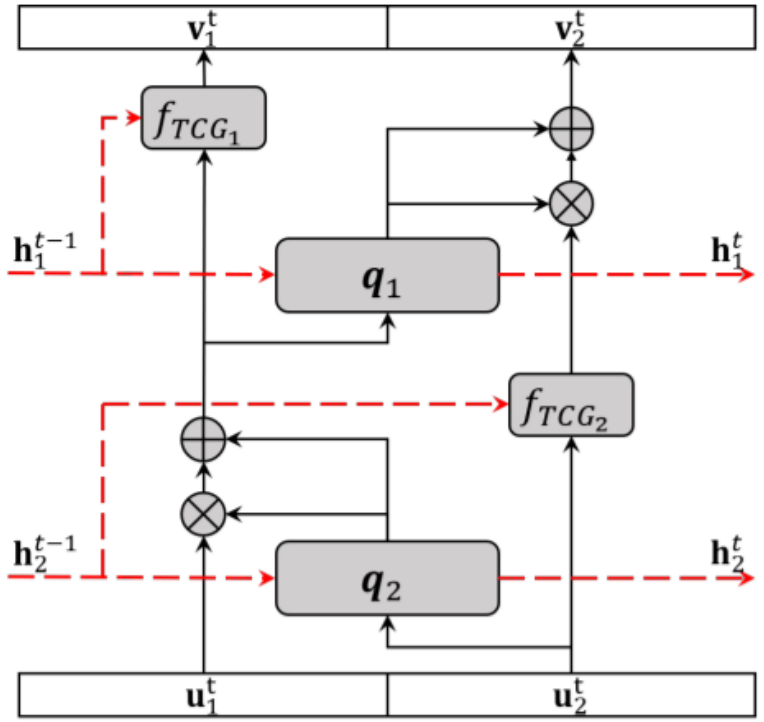


Figure 3: The CRow model. Only the forward map of a single block (two coupling layers) is shown for brevity. The inverse map involves a similar order of operations (analogous to Fig. 2a and Fig. 2b)

Conditional recurrent flow: **conditional** generation of longitudinal samples with applications to neuroimaging